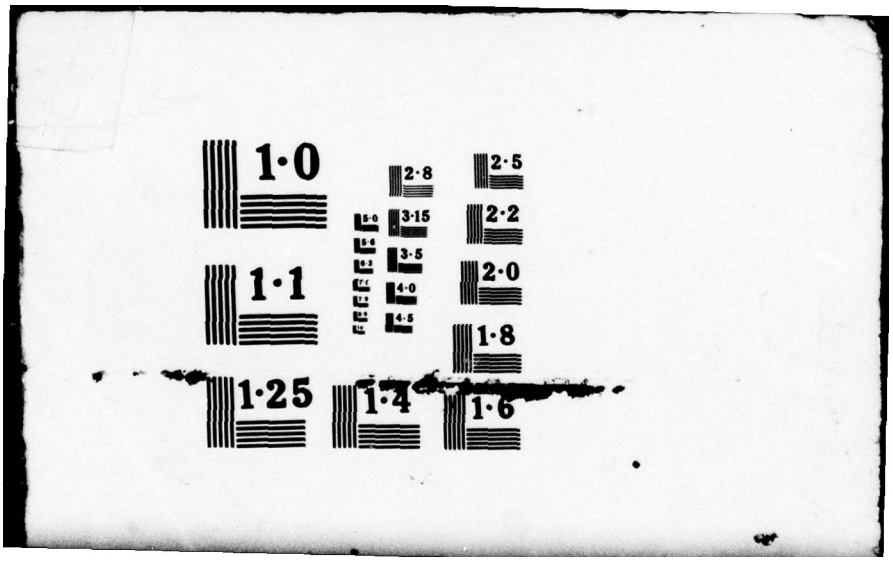
AD-A067 511 GENERAL ELECTRIC CO SYRACUSE N Y HEAVY MILITARY EQUI--ETC F/G 9/5
AN ITERATIVE INTERPOLATION TECHNIQUE USING FINITE IMPULSE RESPO--ETC(U)
FEB 79 J P COSTAS

UNCLASSIFIED R79EMH1 NL

1 OF 2
ADA
067511


1·0
1·1
1·25
1·4
1·6
1·8
2·0
2·2
2·5
2·8
3·15
3·5
4·0

(5) LEVEL II

GENERAL ELECTRIC

HEAVY MILITARY EQUIPMENT DEPARTMENT

(9) TECHNICAL INFORMATION SERIES

(10) Author: J. P. Costas

Subject Category: Digital Signal Processing

(11) No. R79EMH1

Date: Feb 1979

(12) 104 p.

(6) Title: AN ITERATIVE INTERPOLATION TECHNIQUE USING FINITE IMPULSE RESPONSE DIGITAL FILTERS

Copies Available at:
HMED TIS Distribution Center
Box 4840 (CSP 4-18)
Syracuse, New York 13221

GE Class: 1

Govt Class: Unclassified

No. of Pages: 104

Summary

A highly efficient iterative interpolation technique is described in which very high sampling rate multiplication ratios may be obtained without difficulty.

This work leads to a filter design procedure in which the essential features of the passband may be scaled in frequency by any desired factor. A "stretch and fill" tactic is used in the time domain to provide passband frequency scaling and to maintain stopband suppression. Computation of the coefficients for these filters, of any order, is trivial.

A FORTRAN computer program is provided for general interpolation service. Auxiliary use of this program for obtaining filter coefficients is described, and examples of filter synthesis are given. Special entry points are provided for data format mode conversion: complex-to-real and real-to-complex.

Equivalent procedures for sampling rate reduction are also presented and discussed.

GENERAL ELECTRIC COMPANY TECHNICAL INFORMATION

Within the limitations imposed by Government data export regulations and security classifications, the availability of General Electric Company technical information is regulated by the following classifications in order to safeguard proprietary information:

CLASS 1: GENERAL INFORMATION

Available to anyone on request.
Patent, legal and commercial review required before issue.

CLASS 2: GENERAL COMPANY INFORMATION

Available to any General Electric Company employee on request.
Available to any General Electric Subsidiary or Licensee subject to existing agreements.
Disclosure outside General Electric Company requires approval of originating component.

CLASS 3: LIMITED AVAILABILITY INFORMATION

Original Distribution to those individuals with specific need for information.
Subsequent Company availability requires originating component approval.
Disclosure outside General Electric Company requires approval of originating component.

CLASS 4: HIGHLY RESTRICTED DISTRIBUTION

Original distribution to those individuals personally responsible for the Company's interests in the subject.
Copies serially numbered, assigned and recorded by name.
Material content, and knowledge of existence, restricted to copy holder.

GOVERNMENT SECURITY CLASSIFICATIONS, when required, take precedence in the handling of the material. Wherever not specifically disallowed, the General Electric classifications should also be included in order to obtain proper handling routines.

GENERAL ELECTRIC COMPANY
HEAVY MILITARY EQUIPMENT DEPARTMENT
TECHNICAL INFORMATION SERIES

SECTION Advanced Development Engineering

UNIT

HMED ACCOUNTING REFERENCE 570

COLLABORATORS

APPROVED J. C. Buchta TITLE Mgr ADE LOCATION CSP 4-58

R79EMH1

MINIMUM DISTRIBUTION - Government Unclassified Material (and Title Pages) in G.E. Classes 1, 2, or 3 will be the following.

Copies	Title Page Only	To
0	1	Legal Section, HMED (Syracuse)
0	1	Manager, Technological Planning, HMED (Syracuse)
5	6	G-E Technical Data Center (Schenectady)

MINIMUM DISTRIBUTION - Government Classified Material, Secret or Confidential in G.E. Classes 1, 2, or 3 will be the following.

1	0	Manager, Technological Planning, HMED (Syracuse)

ADDITIONAL DISTRIBUTION (Keep at minimum within intent of assigned G.E. Class.)

COPIES	NAME	LOCATION
5 (CLASS 1 ONLY)	DEFENSE DOCUMENTATION CENTER	CAMERON STATION, ALEXANDRIA, VA. 22314
1	L. I. Chasen	P. O. Box 8555 Philadelphia, Pa., 19101
1	W. B. Adams	CSP 4-48
1	S. P. Applebaum	CSP 4-38A
1	L. W. Bauer	CSP 4-4
1	D. L. Beall	FRP 1-J7
1	A. P. Belleisle	EP 3-102
1	R. D. Berlin	FRP 1-5F
1	R. B. Bockstiegel	CSP 5-H4
1	C. Brown	EP 3-144
1	J. C. Buchta	CSP 4-58
1	E. K. Chin	CSP 4-4
1	B. H. Cole	CSP 5-H1
1	T. E. Connolly	CSP 4-57
1	*G. R. Cooper	(Purdue)
1	J. P. Costello	EP3-234
1	F. R. Dickey, Jr.	CSP 4-38A
1	M. D. Egtvedt	EP 3-220

*Cooper - Professor, School of Electrical Engineering, West Lafayette, Indiana 47907

HMED 4024 REV. 3/21/78

A

Copies	Name	Location
1	J.W. Fauwcett	EP 3-112
1	M.M. Fitelson	EP 3-218
1	S.M. Garber	CSP 4-48
1	B.H. Geyer, Jr.	CSP 5-T1
1	J.M. Geyer	EP 3-227
1	**H.W. Hadley	(SRC)
1	M. Hunter	MD 289 French Road Plant Utica, New York 13503
1	M.H. Hutchinson	FRP 1-N1
1	J.M. Irwin	EP 3-140
1	M.A. Johnson	CSP 4-58
1	T.G. Kincaid	Bldg 37, Room 523 Schenectady, New York
1	L. Knickerbocker	FRP 1-N1
1	W.W. Knight	EP3-225
1	J.P. Lindhuber	CSP 4-48
1	T.R. Loeffler	FRP 1-4M
1	J.F. Lomber	FRP 1-C1
1	J.W. Lunden	EP 3-201
1	W.T. Mandeville	FRP 1-N1
1	M.B. McGuiness	FRP 1-N1
1	D.R. Morgan	EP 3-234
1	C.E. Nelson	CSP 4-38A
1	R. Nitzberg	CSP 4-5
1	G. Oehling	CSP 4-5
1	K.A. Olsen	CSP 4-57
1	A.V. Oppenheim	MIT, Dept of EE & Comp Science, Rm 36-625, Cambridge, Ma 02139
1	W.A. Penn	EP 3-221
1	D.W. Perkins	CSP 5-T2
1	N.R. Powell	EP 3-142
1	C.M. Puckette	Bldg 37, Rm 449 Schenectady, New York
1	L.R. Rabiner	Bell Labs, Rm 2D533 Murray Hill, New Jersey 07974
1	J.A. Raper	EP 3-115
1	D.C. Rollenhagen	EP 3-116
1	J.A. Rougas	CSP 4-4
1	F.J. Ryan	CSP 5-H4
1	H. Scudder	Bldg 5, Rm 209 Schenectady, New York
1	F.D. Shapiro	CSP 4-5
1	T.B. Shields	CSP 3-4

**Hadley - Merrill Lane, Syracuse, New York

Copies	Name	Location
1	O. H. Shuart	CSP 4-38B
1	J. F. Slocum	CSP 4-4
1	M. T. Stark	CSP 4-48
3	C. A. Stutt	Bldg 5, Rm 209 Schenectady, New York
1	B. W. Tietjen	FRP 1-N1
1	B. P. Viglietta	CSP 4-57
1	A. H. Vural	CSP 4-48
1	E. C. Wert	CSP 3-20A
1	W. P. Whyland	CSP 4-48
1	J. B. Williams	EP 3-225
1	K. F. Williams	CSP 4-5
1	D. W. Winfield	CSP 4-48
1	J. A. Ladstatter	FRP 1-N
1	D. T. Hurley	MD 310 French Road Plant Utica, New York 13503
1	J. A. Lash	U 2420 P. O. Box 8555 Philadelphia, Pa. 19101
1	J. J. Foley	U 4424 P. O. Box 8555 Philadelphia, Pa. 19101
1	E. Niemann	U 4424 P. O. Box 8555 Philadelphia, Pa. 19101
1	J. Bondy	U 1219 P. O. Box 8555 Philadelphia, Pa. 19101

C

ABSTRACT

Interpolation operations have wide applications in signal processing systems. An efficient interpolation technique is developed and demonstrated. This is an iterative technique in which interpolation output at any stage is combined with the input to that stage so that the interleaved result forms the input to the succeeding stage. Sampling rate increases of high order are not difficult to achieve. Resampling of high-order interpolator outputs permits precise control of sampling rate conversion ratios.

From this basic work in interpolation, a digital filter synthesis procedure is developed. The resulting class of finite impulse response filters is competitive in speed and storage with conventional designs. These filters have the property that the passband frequency function of a generic design can be scaled by any desired factor in frequency while maintaining stopband suppression to specified levels. This synthesis procedure uses a "stretch-and-fill" iteration operation. The impulse response into a synthesis stage is stretched, and the vacated spaces are then filled from new data synthesized at that stage. The "stretch" performs the frequency scaling function while the "fill" operation eliminates image responses that would otherwise appear. Computation of the coefficients for these filters, of any order, is trivial.

A FORTRAN computer program which uses the developed algorithm is provided for general interpolation service. Auxiliary use of this program for obtaining derived filter coefficients is described, and examples of filter synthesis are given. Special entry points are provided for data format mode conversion: complex-to-real and real-to-complex.

Equivalent procedures for sampling rate reduction are also presented and discussed.

ACKNOWLEDGMENTS

The writer is pleased to acknowledge the helpful comments and criticisms provided by Mr. S. P. Applebaum, Dr. L. W. Bauer, Dr. F. R. Dickey, Jr. and Dr. W. P. Whyland, all of the General Electric Company.

Special thanks are due Dr. L. R. Rabiner of the Bell Telephone Laboratories who brought the writer's attention in a recent conversation to the material cited in references five through ten. Even though the specific procedures and key results of this paper have not been anticipated, prior knowledge of these references would have permitted better citation of related work.

I am also indebted to Mr. H. W. Hadley of the Syracuse Research Corporation for making available some excellent survey material covering the field of digital signal processing. These reports were written by Mr. Hadley as internal company memos before his recent departure from the General Electric Company.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Continued)

SECTION I

INTRODUCTION

The basic technique of interpolation is familiar to most engineers in the context of "reading between" tabulated points in tables of functions or other data. This may be an unfortunate introduction to a useful, fascinating, and often overlooked technique for reasons that go beyond the immediate mental anguish of doing the proportions correctly.

Satisfactory linear interpolation requires a fineness in the spacing of the abscissa values or "sampling rate" that is often far greater than that required to contain the basic behavior information of the function or data. This is especially true in digital signal processing systems where band-limited functions having accurately known spectra are involved. In such systems economies in design are sometimes overlooked because changes in the sampling rate within the system would be required. Such changes are most conveniently accomplished by the resampling of interpolated data. Such interpolation must be done inexpensively using designs which allow control of introduced errors.

In some cases sampling rates that are adequate to permit reconstruction of the continuous waveform are inadequate for certain nonlinear operations, with multiplication being a common example. In these cases interpolation may be used to raise the sampling rate just prior to a nonlinear operation of this type.

Interpolation is obviously not confined to the time domain. In the spatial domain for example, expensive beamformer processors can sometimes be saved by interpolation-synthesis of outputs corresponding to intermediate pointing angles.

Data format or mode conversion between "real" data and "complex" data is often required in signal processing work. This conversion problem can be shown to have a simple interpolation solution. Media conversions often involve interpolation because of differences in appropriate sampling rates. Samples of a waveform taken at or near the Nyquist rate (two times the highest signal frequency) might be adequate for machine processing. However this rate would be totally inadequate for display use. Try, for example, to mentally reconstruct a sine wave given, say, three-equispaced randomly located sample values taken from one period. Interpolation prior to display is often a necessity.

In this paper an interpolation technique is developed which is basically quite simple and is very easy to use. The technique appears to offer computational advantages over more conventional interpolation methods. A modification of the interpolation process leads to a filter synthesis procedure which is described. Applications such as mode transformations are treated in detail. A computer program for interpolation and mode conversion is presented along with instructions for use.

The basic techniques developed for interpolation may also be used for sampling-rate reduction. Computer programs and design procedures are presented which permit convenient sampling rate conversions either up or down.

SECTION II

BASIC SINGLE-LEVEL INTERPOLATION

2.1 FREQUENCY DOMAIN APPROACH TO INTERPOLATION

For present purposes it is convenient to formulate the interpolation problem in the context of the tapped delay line representation of Figure 2-1. A sinusoidal signal

$$s(t) = \cos 2\pi ft \tag{2-1}$$

is applied to a delay line having 2N taps spaced T-seconds apart. In later development T will represent a sampling period for a sampling rate R where

$$R = \frac{1}{T} \tag{2-2}$$

It is desired that the output at the center of the line be obtained where

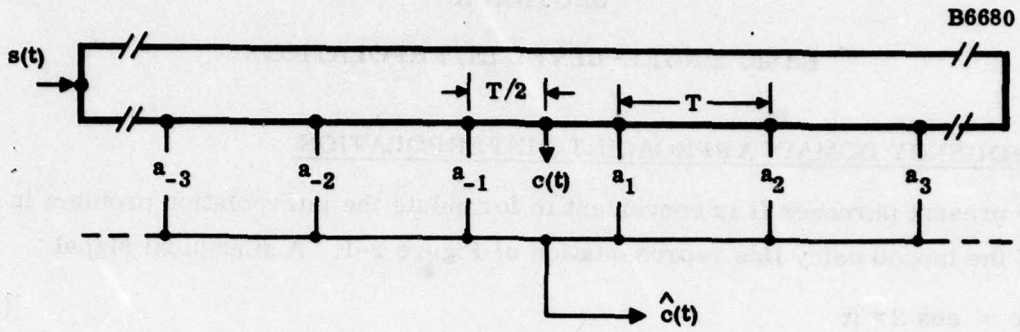
$$c(t) = \cos 2\pi f\left[t - (2N-1)\frac{T}{2}\right] \tag{2-3}$$

An estimated center output $\hat{c}(t)$ is to be delivered from the weighted sum of the tap voltages so that

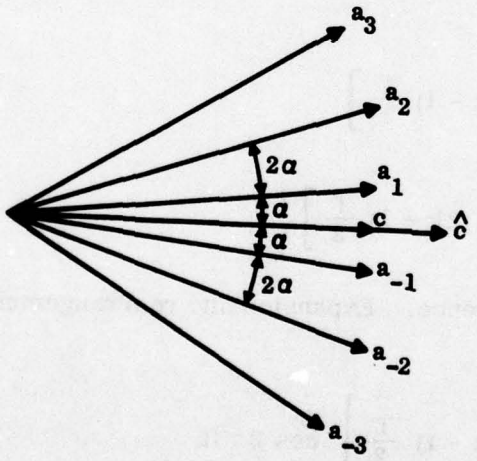
$$\hat{c}(t) = \sum_{k=1}^{N} \left\{ a_k \cos 2\pi f\left[t + (2k-1)\frac{T}{2}\right] + a_{-k} \cos 2\pi f\left[t - (2k-1)\frac{T}{2}\right]\right\} \tag{2-4}$$

using the signal at the center tap as reference. Expansion and rearrangement of the terms in Equation (2-4) yields

$$\hat{c}(t) = \left[\sum_{k=1}^{N} (a_k + a_{-k}) \cos 2\pi f(2k-1)\frac{T}{2}\right] \cos 2\pi ft - \left[\sum_{k=1}^{N} (a_k - a_{-k}) \sin 2\pi f(2k-1)\frac{T}{2}\right] \sin 2\pi ft \tag{2-5}$$



a. Tapped Delay Line Representation



$$\alpha = 2\pi \frac{fT}{2} = \frac{\pi}{2}\left(\frac{f}{\frac{R}{2}}\right)$$

b. Phasor Diagram

Figure 2-1. Interpolation Processor, Continuous Time Version

Since the center tap signal was used as reference for Equations (2-4) and (2-5), we may ignore the (2 N - 1) T/2 delay term of Equation (2-3) and represent cos 2π ft as the desired output signal. It then becomes apparent that the first term of Equation (2-5) represents an in-phase component while the second term represents a quadrature component relative to the desired cos 2 π ft signal. In order to eliminate the undesired quadrature term choose:

$$a_k = a_{-k} \tag{2-6}$$

and define

$$\left.\begin{aligned} b_k &= 2\, a_k \\ a_k &= \frac{b_k}{2} \end{aligned}\right\} \tag{2-7}$$

So that Equation (2-5) becomes after Equations (2-2), (2-6) and (2-7) are used

$$\hat{c}(t) = \left[\sum_{k=1}^{N} b_k \cos (2k-1)\,\frac{\pi}{2}\,\frac{f}{\left(\frac{R}{2}\right)}\right] \cos 2\pi\, ft \tag{2-8}$$

The estimate of Equation (2-8) has no phase error so that the bracketed factor may be interpreted as the real filter transfer function H (f) of the estimator of Figure 2-1a (time delay of (2 N - 1) T/2 understood) where

$$H(f) = \sum_{k=1}^{N} b_k \cos (2k-1)\,\frac{\pi}{2}\,\frac{f}{\left(\frac{R}{2}\right)} \tag{2-9}$$

If H (f) were equal to unity, the estimation would be perfect. As shown in Figure 2-1b the coefficient choice of Equation (2-6) insures that the resultant phasor $\hat{c}$ lies along the true center phasor line c. However, the amplitude of $\hat{c}$ may differ from unity and hence introduce estimation errors. The problem then becomes one of choosing the coefficients b_k so that H (f) of Equation (2-9) remains as close as possible to unity over a frequency range which will be confined to

$$0 \le f < \frac{R}{2} \tag{2-10}$$

Both the phasor diagram of Figure 2-1b and Equation (2-9) show the difficulty involved, namely that the component angular relationships are functions of frequency. At zero frequency the phasors are all collinear, and the sum of the coefficients should be set equal to unity. As the frequency is increased the phase relationships change and a different coefficient set b_k is required for unity gain. Since the single frequency analysis is meant to represent performance at only one of a band of signal frequencies, the b_k choice cannot be changed as a function of frequency. An optimum b_k set (in some sense) must be found for the signal characteristics of the application.

2.2 PROCESSOR COEFFICIENT DETERMINATION

Examination of Equation (2-9) reveals that an odd harmonic series of cosine terms is involved with each cosine term having unit value at f = 0 and zero value at f = (R/2). Thus H (f = R/2) = 0 regardless of the b_k set chosen. Define now a maximum value of f/(R/2) and consider design procedures confined to the range

$$0 \leq \frac{f}{\left(\frac{R}{2}\right)} \leq \gamma \quad . \tag{2-11}$$

It is desired to minimize the peak error (deviation from unity) of H(f). If the error is defined as δ then the quantity to be minimized is δ_{max} where

$$\delta = \left|1 - H(f)\right| \tag{2-12}$$

That is, we seek that set $\{b_k\}$ which minimizes the maximum value of δ of Equation (2-12) for a given γ, N. As expected, the computational load (required N value) is increased when δ_{max} is made smaller and when γ is made larger (closer to unity).

Consider now the trivial but illustrative case in which N = 1. H(f) now becomes

$$H(f) = b_1 \cos \frac{\pi}{2} \frac{f}{\frac{R}{2}} \tag{2-13}$$

When f = 0, H(f) is equal to b_1 and when $f/(R/2) = \gamma$, $H(f) = b_1 \cos \pi\gamma/2$. Since the H(f) behavior is monotonic, b_1 must be chosen to equalize the errors at the frequency extremes. The choice is clearly

$$b_1 = \frac{2}{1 + \cos \frac{\pi\gamma}{2}} \tag{2-14}$$

and

$$\delta_{max} = \frac{1 - \cos \pi\gamma/2}{1 + \cos \pi\gamma/2} \tag{2-15}$$

which may be rewritten as

$$\cos \pi\gamma/2 = \frac{1 - \delta\ max}{1 + \delta\ max} \tag{2-16}$$

The interaction of performance and bandwidth for this simple case is easily calculated. For δ_{max} levels of -20, -40, and -60 dB, the respective γ values are 0.39, 0.13, 0.04. Thus for a fixed computation load (N = 1 in this case) one can only trade performance for bandwidth.

The above procedure represents a modified form of linear interpolation using data from the delay line tap pair adjacent to the center point of Figure 2-1a. Ordinary linear interpolation would involve (a_1, a_{-1},) values of 1/2 and a unit value of b_1. Note that b_1 from Equation (2-14) is always greater than one. This departure from ordinary linear interpolation is necessitated by the desire to minimize the peak error in H(f) over the operating bandwidth. The result in this case is a 6 dB reduction in peak error level as compared to the normal linear interpolation usage of the tap signals.

The computational approach used for general $\{b_k\}$ determination will now be described. First rewrite Equation (2-9) letting

$$x = \frac{f}{R/2} \tag{2-17}$$

so that

$$H(x) = \sum_{k=1}^{N} b_k \cos (2k - 1) \frac{\pi}{2} x \tag{2-18}$$

In the x interval from 0 to γ select a set of equally spaced points $\{x_i\}$ and consider the weighted squared error sum

$$E = \sum_i w_i \left[1 - H(x_i) \right]^2 \tag{2-19}$$

The process starts with unit weights w_i and a $\{b_k\}$ set is determined by the usual (Ref. 1) least squares procedures in which

$$\cos (2k - 1) \frac{\pi}{2} x$$

is used to define the coordinate functions of the approximation. After the least-squares operation is complete, the weights w_i are modified in proportion to

$$\left|1 - H(x_i)\right|$$

so that the regions of high error are emphasized. Then a new weighted least-squares fit procedure is invoked to derive a modified coefficient set $\{b'_k\}$. After a few such iterations it will be found that the peak errors of H(x) over the approximation region are equalized and, coincidentally, minimized. (Several years ago the author wrote an interactive time-share program for this and a variety of other curve-fitting problems which leans heavily on a Honeywell H-635 auxiliary library routine named LSQMM. The names M.A. Martin and F.E. Liller are associated with this program.) This procedure appears to have more general applicability and may be found more convenient to use than the REMEZ exchange algorithm employed for similar purposes (Ref. 2)

These numerical procedures were used to obtain the performance data shown in Figure 2-2. The particular cases displayed here were run for specific purposes to be described in later sections. For the present, the results of Figure 2-2 may be discussed in terms of the general interpolator design problem.

The time-share program previously mentioned was run for a variety of γ values over an appropriate set of N values. For each N value the peak error in H(x) was found and expressed in decibels. Plots of δ_{max} in dB as a function of N were then made using δ as a parameter. For a specific γ, say 80%, the peak error decreases monotonically with N as expected. For this particular γ only, the error performance for least-squares determination of the coefficient set $\{b_k\}$ is shown circled above and to the right of the normal minmax performance curve marked "γ = 80%". It appears that the minmax iteration is worth the order of 6 to 8 dB in additional error suppression.

The rapidly escalating processor costs incurred as γ approaches one may be seen by looking along the -35 dB error line, for example. At γ = 80% an N of 6 will suffice, at γ = 90% an N of 11 is indicated, while at γ = 95% the computation price has risen to N = 23. For the smaller γ values N goes down rapidly, with simple two-point interpolation (N = 1) providing error performance in excess of -50 dB for γ of values of about 15% and less.

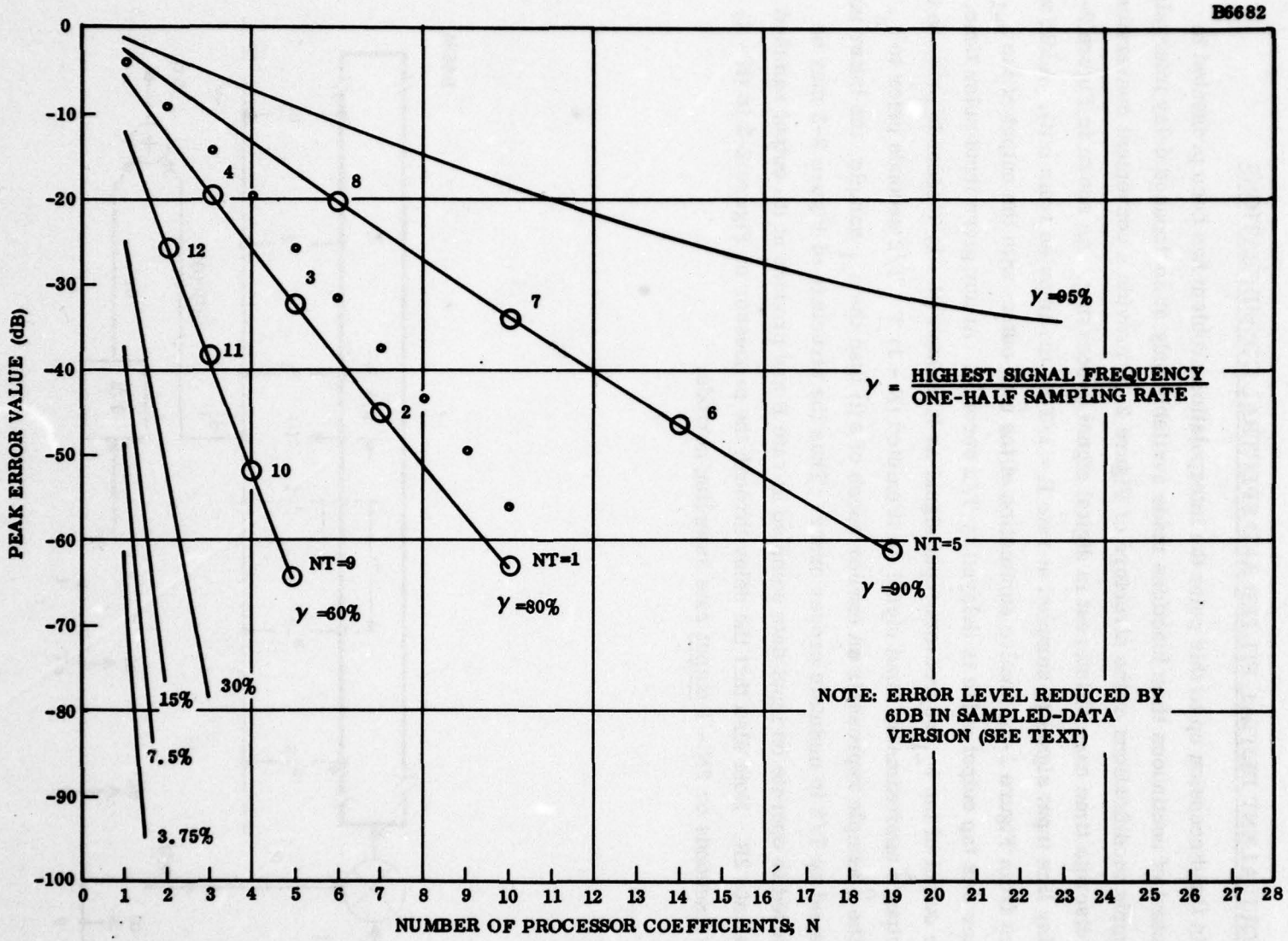


Figure 2-2. Basic Single Level Interpolator Performance Curves

2.3 EQUIVALENT DIGITAL FILTER AND SPECTRAL CONSIDERATIONS

In the discussion up to this point the interpolation problem has been presented in the context of continuous time functions made available only at equispaced delay intervals T. Simple modifications of the structure of Figure 2-1a provide a convenient conversion to the discrete time case of interest in digital signal processing. As shown in Figure 2-3 the delay line input signal is sampled at rate R = 1/T by the impulse train p (t). Additional changes from Figure 2-1a involve summation of the $\hat{c}$ (t) output with the output of the a_{-1} tap after this tap output pulse is delayed by T/2 seconds. At any given input pulse time, t_n, the output of tap a_{-1} represents input signal as it existed (N - 1) T seconds prior to t_n. The output $\hat{c}$ approximates input signal as it existed (N - 1) T + T/2 seconds prior to t_n. Since the $\hat{c}$ sample represents an earlier epoch of s (t) than the a_{-1} sample, the latter must be delayed by T/2 to maintain proper order. Thus the structure of Figure 2-3 may be considered to operate on input data sampled at rate R and produce at its output sampled data at rate 2R. Note also that the delay through the processor of Figure 2-3 is (N - 1) T + T/2 seconds or 2N - 1 output rate sampling periods.

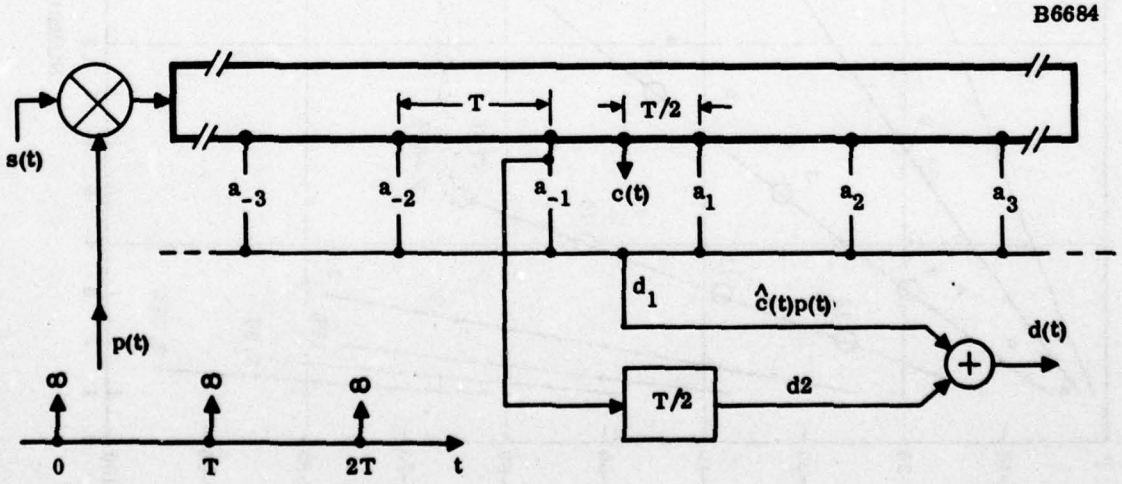


Figure 2-3. Interpolation Processor, Sampled Data Version

In the analysis of Figure 2-3 which follows it is assumed that s (t) has a power density spectrum S (f) where

$$S(f) = 0 \text{ for } |f| \geq R/2 \tag{2-20}$$

Since the sampling period is precisely equal to the delay line tap spacing, all line output voltages will be zero except for times which are multiples of the sampling period T. The tap voltage d_1 (t) will be

$$d_1(t) = \hat{c}(t)\, p(t) \tag{2-21}$$

The output on the a_{-1} tap represents signal data that is from a later time than the data on the $\hat{c}(t)$ bus by T/2 seconds. For this reason the a_{-1} impulse samples of Figure 2-3 must be delayed as shown in order to maintain proper output sequence. This T/2 delay of the a_{-1} sequence is exactly equivalent to

$$d_2(t) = s(t - (2N - 1)\, T/2)\, p(t - T/2) \tag{2-22}$$

The impulse train p (t) may be represented by the Fourier series

$$p(t) = \sum_{n=0}^{\infty} \cos 2\pi\, nR\, t \tag{2-23}$$

so that

$$P(t - T/2) = \sum_{n=0}^{\infty} (-1)^n \cos 2\pi\, nRt. \tag{2-24}$$

Equation (2-22) may be written as

$$d_2(t) = s(t - (2N - 1)\, T/2) \sum_{n=0}^{\infty} (-1)^n \cos 2\pi\, nRt \tag{2-25}$$

and the other adder input will be

$$d_1(t) = \hat{c}(t)\, p(t) = \hat{c}(t) \sum_{n=0}^{\infty} \cos 2\pi\, nRt. \tag{2-26}$$

Combining gives

$$d(t) = \left[\hat{c}(t) + s(t - 2(N-1)T/2) \right] \sum_{n=0,2,4}^{\infty} \cos 2\pi nRt$$

$$+ \left[\hat{c}(t) - s(t - 2(N-1)T/2) \right] \sum_{n=1,3,5}^{\infty} \cos 2\pi \cos 2\pi nRt \qquad (2\text{-}27)$$

It is trivial to show that if x (t) has a power density spectrum X (f), then

$$Y(f) = \frac{1}{4} \left[X(f - f_c) + X(f + f_c) \right] \qquad (2\text{-}28)$$

where

$$y(t) = x(t) \cos 2\pi f_c t \qquad (2\text{-}29)$$

and Y (f) is, of course, the power density spectrum of y (t). The result in Equation (2-28) will now be used with Equation (2-27) to derive the positive frequency portion of the spectrum of d (t), $D^+(f)$.

From the discussion relative to Equation (2-9), the transfer function relating $\hat{c}$ (t) and s (t) is H (f) exp (- j 2πf (2N - 1) T/2). This allows use of Equation (2-28) so that the power density spectrum $D^+(f)$ of d(t) of Equation (2-27) may be written as (factor of 1/4 ignored)

$$D^+(f) = \sum_{n=0,2,4}^{\infty} |1 + H(f - nR)|^2 \, S(f - nR)$$

$$+ \sum_{n=1,3,5}^{\infty} |1 - H(f - nR)|^2 \, S(f - nR) \qquad (2\text{-}30)$$

A sketch of Equation (2-30) for an S(f) of unity in the range $|f| \leq R/2$ is shown in Figure 2-4a. The sampled data spectrum at the input to the interpolator has a periodicity of frequency span R, the input sampling rate. The spectral periodicity at the interpolator output has a span 2R consistent with the doubling of the sampling rate as explained relative to Figure 2-3. Thus the new foldover frequency (one-half the sampling rate) is R.

Relative to this new foldover frequency we may define an effective frequency function G(f) for the interpolation processor of Figure 2-3

$$\left.\begin{aligned} G(f) &= |1 + H(f)| \quad , \quad 0 \leq f \leq R/2 \\ G(f) &= |1 - H(R - f)| \, , \quad R/2 \leq f \leq R \end{aligned}\right\} \tag{2-31}$$

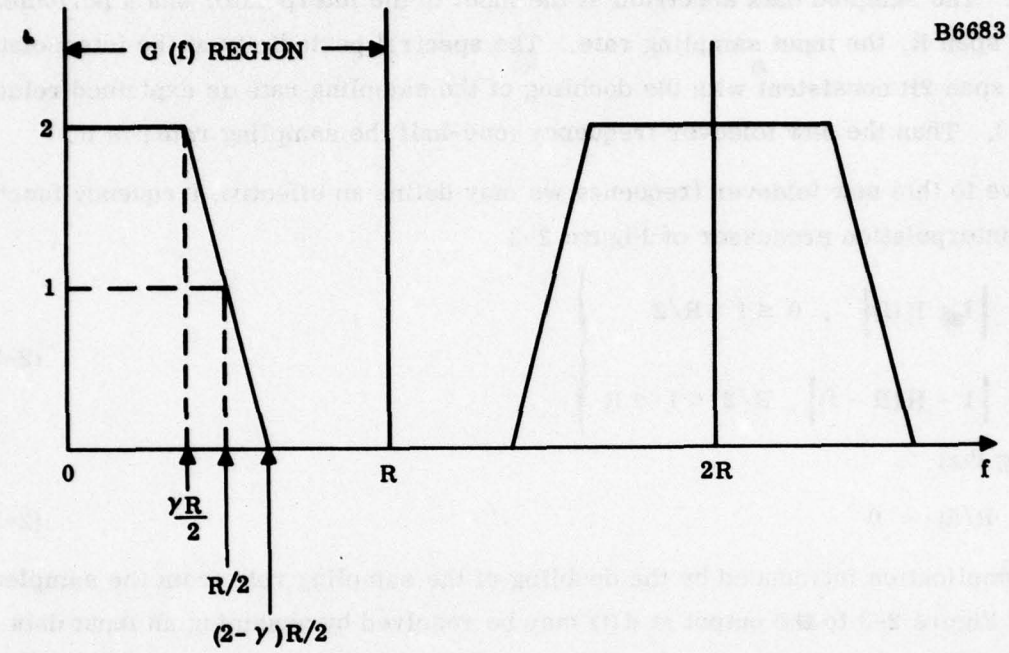
remembering that

$$H(f = R/2) = 0 \tag{2-32}$$

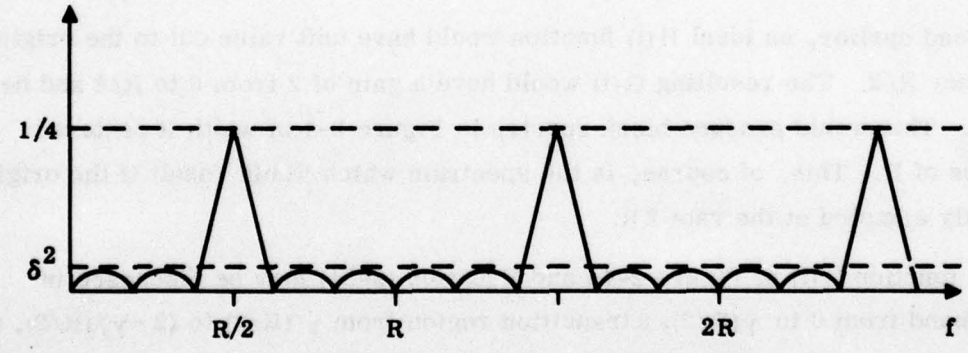
The complication introduced by the doubling of the sampling rate from the sampled data input of Figure 2-3 to the output at d(t) may be resolved by assuming an input data rate of 2R with alternate sample values all equal to zero (Ref. 3). This construction leaves the physical input signal to the line of Figure 2-3 unchanged, yet allows input and output power density spectra to be directly compared in the context of an equivalent filter G(f) of Equation (2-31).

As discussed earlier, an ideal H(f) function would have unit value out to the original foldover frequency R/2. The resulting G(f) would have a gain of 2 from 0 to R/2 and be zero elsewhere. This would produce block spectra in Figure 2-4 of width R centered at even multiples of R. This, of course, is the spectrum which would result if the original s(t) were directly sampled at the rate 2R.

The filter function G(f) of Figure 2-4a and Equation (2-31) may be discussed in terms of a passband from 0 to $\gamma(R/2)$, a transition region from γ (R/2) to $(2-\gamma)(R/2)$, and a stopband from $(2 - \gamma)(R/2)$ to R. The passband and stopband are directly related since Equation (2-31) shows that a deviation δ of H(f) from unity in the passband at frequency f results in a response of magnitude $|\delta|$ in the stopband at frequency R - f. Interpolation errors result in the desired signal being at an incorrect level at the output (passband) accompanied by an image signal created within the processor (stopband).



a. Interpolator Output Spectrum



b. Interpolator Error Spectrum

Figure 2-4. Interpolator Output and Error Spectra

The gain of 2 in the passband indicates a reinforcement of desired output signal from the interpolated series by the signal from the "direct-through" series from tap a_{-1} of Figure 2-3. Remember that the error performance calculations shown in Figure 2-2 were made relative to the interpolated series only. Thus the relative error values of the sampled-data version of Figure 2-3 are reduced by 6 dB from the values given in Figure 2-2.

The transition region of Figure 2-4 is obviously a region of poor performance. The desired signal level errors can be large, resulting in high levels of image-signal generation. Error effects resulting from this region can be reduced either by making γ large (which can be computationally very expensive) or by keeping the signal spectrum in the region γ R/2 to R/2 low. The latter condition results when the sampling rate is increased beyond twice the low-pass filter cutoff frequency when the initial analog-to-digital conversion is made, for example.

Filters of this type are also discussed in reference 5 and are described as "half-band non-recursive" filters.

SECTION III

MULTILEVEL INTERPOLATION

3.1 EXTENSION OF TECHNIQUE TO GENERAL L-LEVEL CASE

If the sampling rate is to be increased by a factor of 2^L, an iteration of the technique of Figure 2-3 may be used to good advantage as shown in Figure 3-1. The general L-level procedure will be outlined first, with detailed design considerations to follow. (Different design approaches to multistage interpolation may be found in references 7, 8 and 10.)

Starting at the top of Figure 3-1, we reason that the interpolated data at the output of level 1 represents the best estimate of intermediate data values obtainable from the allocated processing investment made at that level. Thus it seems eminently reasonable to use these intermediate results as input for the next processing stage.

The design of the next stage (level 2 in this case) may be relaxed as compared to that of level 1 because the relative signal bandwidth has been cut in half by the 1:2 increase in sampling rate at the previous level.

For example, consider an error level line drawn horizontally in Figure 2-2 through the γ = 60, 30, 15, ---% curves. If the signal bandwidth requirement calls for a γ = 60% first level, a proper 1st-level N value may be chosen by noting the intersection of the error line with the curve γ = 60%.

Now at the second level the effective γ is reduced to 30% and a lesser N will be satisfactory which means a lighter processing load as compared to level 1. The next step results in a level 3 γ of 15%, and so on. Eventually the working-level γ is reduced to the point where N = 1 which, in effect, means linear interpolation. At this point (say at the output of level K) the procedure is changed and the final processor simply does linear interpolation of $2^{L-K}-1$ points between each input pair. This procedure is efficient in that only that amount of processing needed for error control is used at each level.

The procedure may obviously be extended to provide arbitrary sampling rate multiplication at the linear interpolation output level.

The problem specifications determine how many Figure 2-3 type processors will be needed before reaching the 2^{L-K} linear interpolation processor. Once the linear interpolation level is reached sampling rate multiples become inexpensive.

The above discussion is conceptually correct, but important design details which were passed over will be considered in the next two sections.

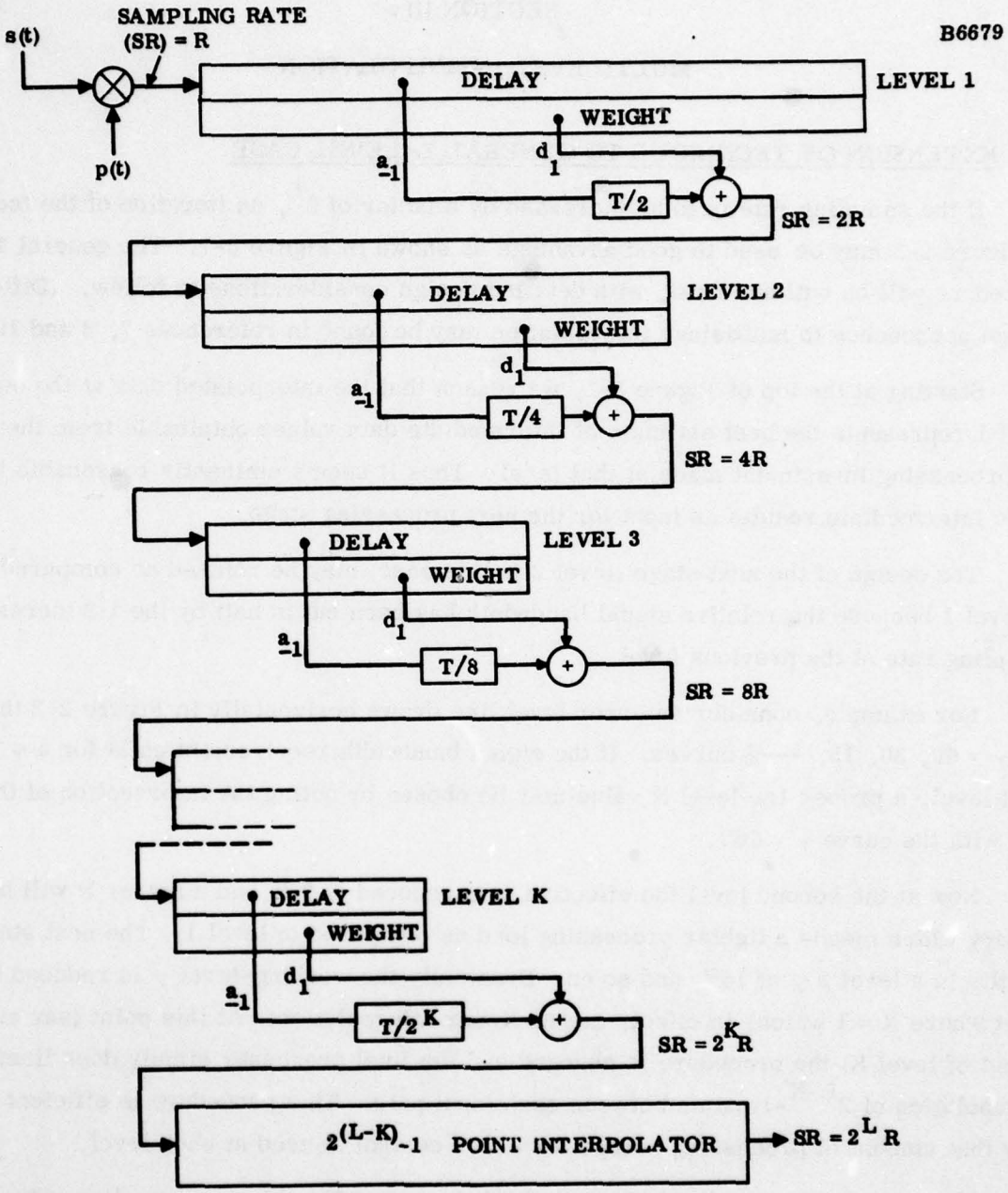


Figure 3-1. Multilevel Interpolation Procedure

This page intentionally left blank.

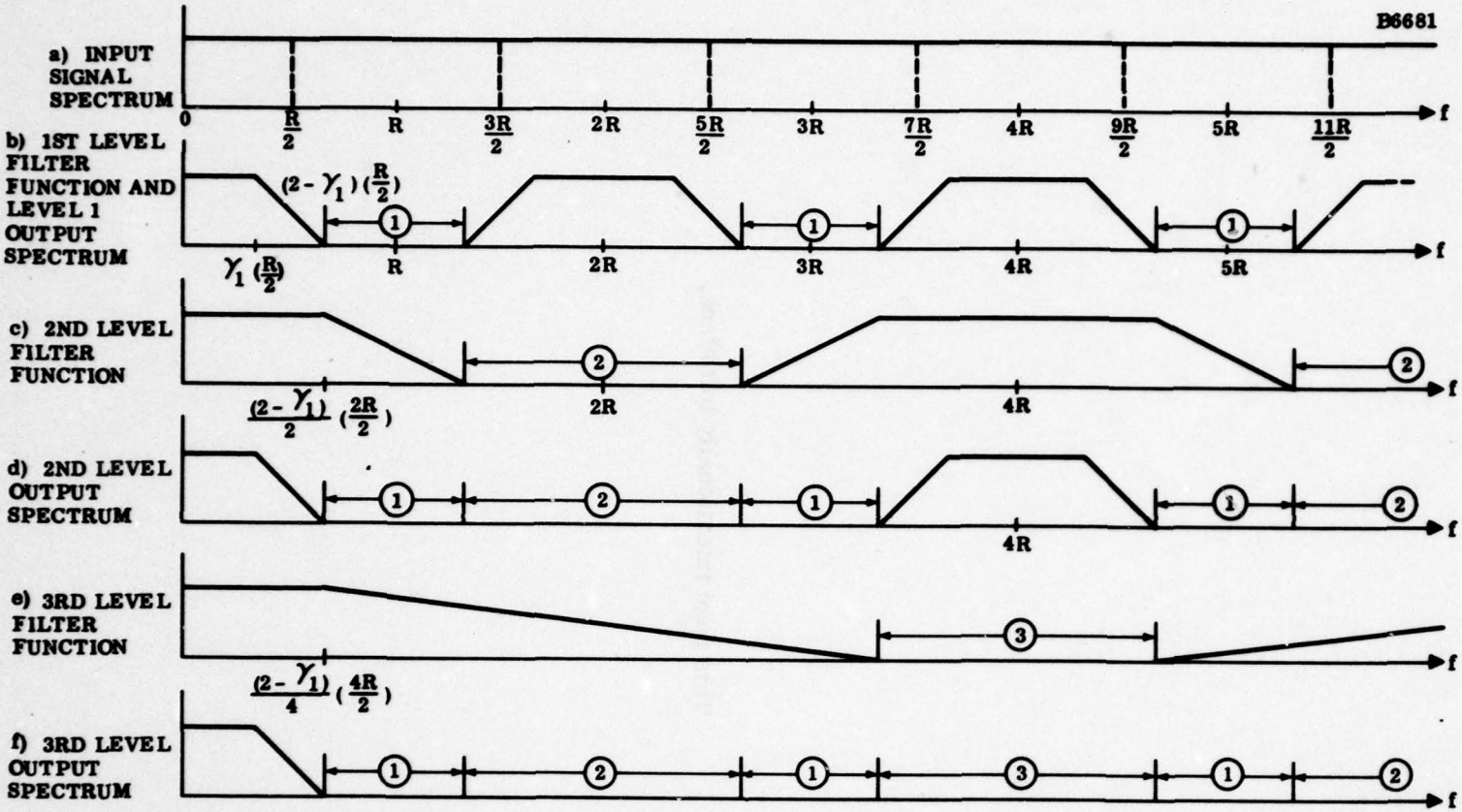


Figure 3-2. System Filter Functions and Spectra, Critically-Sampled Case

3.2 DESIGN CONSIDERATIONS, CRITICALLY SAMPLED CASE

The output spectrum of Figure 2-4a was based on a uniform input signal spectrum extending to R/2. Deviations of this curve from a repeated block spectrum on 2R Hz centers represents error power. A normalized error spectrum is sketched in Figure 2-4b. The peak level δ^2 is reached in the pass and stopbands. In the transition region a high density level is always reached. Error contributions in this region can only be reduced by narrowing the transition region (increasing γ) or by lowering the spectral level of the signal in the foldover-frequency region (oversampling).

Error control in the multilevel interpolator is strongly influenced by the specific application. In much of the previous literature (Ref. 3) interpolation is done by passing a digital signal through a digital low-pass filter which is characterized by a passband, a transition region and a stopband. In order to compare interpolation methods, the design procedures in this section will also be based on an extended stopband in which uniformly low error level is maintained. In the multilevel interpolator, error spectrum distribution can take many forms. One form arises from the use of a low-pass filter in the operation. It is not obvious that this is necessarily the best error distribution for all applications.

Figure 3-2(a) shows a uniform input signal spectrum extending to R/2. Figure 3-2(b) shows the first-level filter function and output spectrum for the critically sampled case. A γ_1 is chosen at the first level which yields a transition region from γ_1 (R/2) to $(2 - \gamma_1)$ R/2 at first-level output. The error spectrum is not shown. However, it is known to peak at the center of each transition region. This first-level design will produce stopbands in the spectrum as shown by the circled number ones, indicating stop regions contributed by level 1.

The level 2 design uses an "extended" γ_2 of $(2 - \gamma_1)/2$ in order that the created stopbands shown as circled two's encompass the transition region and error-spectrum peaks associated with the spectra centered at multiples of 2R. Figure 3-2(c) shows the stopbands produced by the level 2 processor while Figure 3-2(d) shows the level-two output spectrum with the stop regions identified as to the processor level responsible.

Starting with the third level, the γ values can be simply one-half the previous level value as shown in Figure 3-2(e) (since the transition regions of these processors will correspond to previously produced null regions). The spectrum and null region identification after three levels is shown in Figure 3-2(f). Using this design approach, the error in the final output is essentially the error contributed by the first level processor. The first-level error spectrum peak is carefully maintained (unfortunately) in order that all other error peaks in the null region will be suppressed.

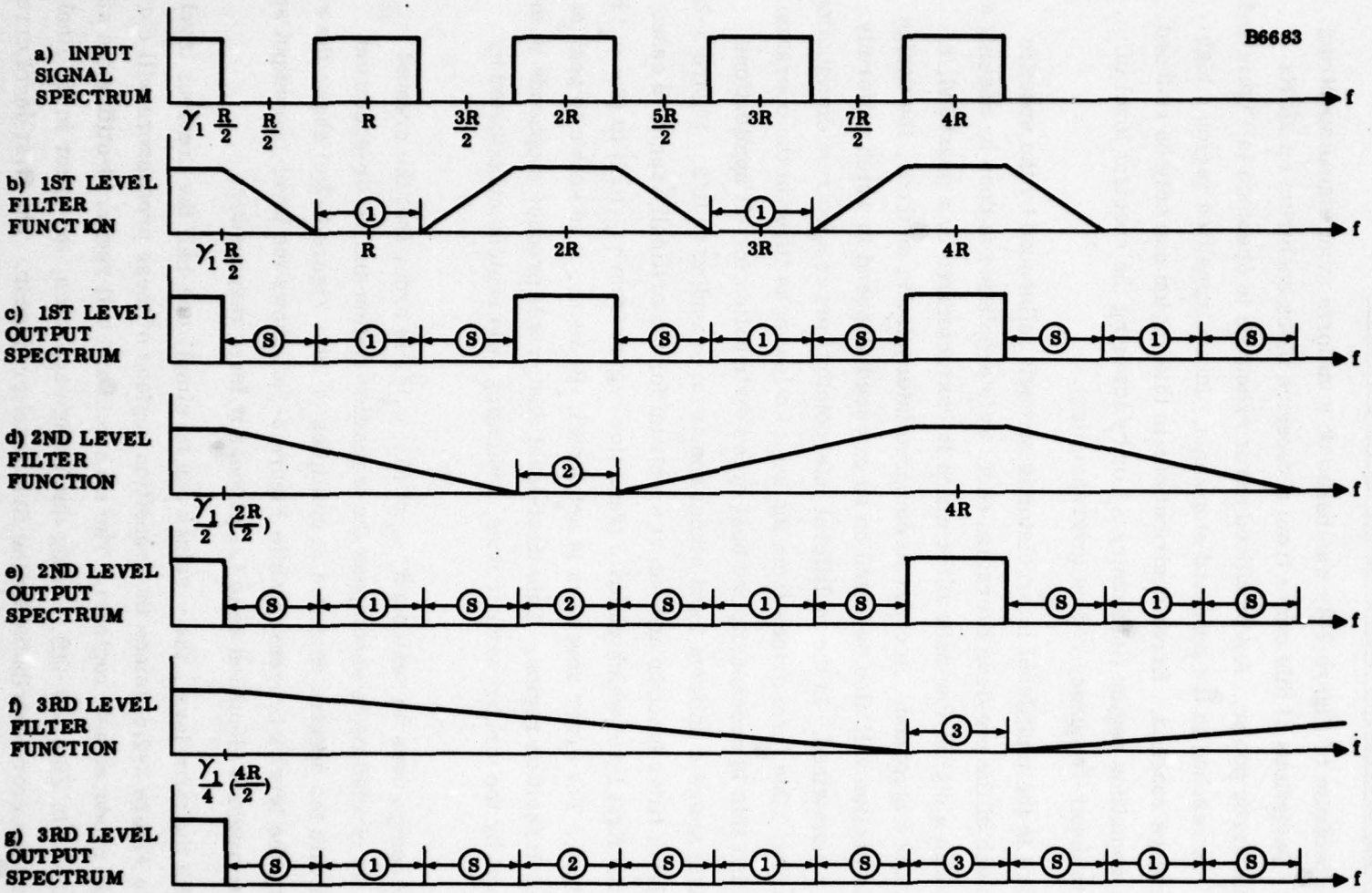


Figure 3-3. System Filter Functions and Spectra, Oversampled Case

3.3 DESIGN CONSIDERATIONS, OVERSAMPLED CASE

A far more pleasant design task is presented when oversampling is done so that a null region in the input spectrum exists as shown in Figure 3-3(a). Now the γ_1 value can be set near the signal cutoff frequency and the γ value can be halved between all levels. The transition region of the first-level processor falls in a null region created by the signal which is identified with a circled S. The remaining plots in Figure 3-3 are self-explanatory based on the Figure 3-2 discussion. There are no error spectrum peaks which need special attention in the oversampled case.

If the signal spectrum does not extend to zero frequency further design economies are possible. In the coefficient determination process described in Section II, par. 2.2, the region over which H(f) is to be held to unity is now confined with "don't care" regions both above and below the passband. This results in better passband performance for a given N value, and corresponds to the "stopband" filtering procedures discussed by Schafer and Rabiner in Reference 3.

In the discussion relative to Figures 3-2 and 3-3 emphasis was placed on the stopbands contributed at each level. The transition region responses have cumulative filtering effects which are significant as will be noted later in the discussion of filter synthesis.

SECTION IV
FILTER SYNTHESIS APPLICATIONS

4.1 PRELIMINARY CONSIDERATIONS

When the interpolation process discussed in Sections II and III is examined as a digital processor and the equivalent frequency function is studied, some interesting results occur. The work in this section leads to a class of digital filters having several interesting properties:

- The impulse response has a central term of unity and has a symmetric distribution about this central value. The resulting finite impulse response (FIR) filter provides a linear phase delay.
- The resulting passband region frequency function can be scaled arbitrarily by stretching the given impulse response and filling in the vacated terms with new values.
- This scaling operation can be extended without limit in the synthesis procedure. The numerical computation involved in coefficient determination is trivial.
- A large variety of filter designs may be obtained from relatively few stored "seed" parameters.
- The designer, in addition to scaling, has control of passband width, transition region width and stopband peak levels. Passband ripple is not directly controllable but is related in a complex way to stopband attenuation. For the usual stopband levels, passband ripple is not a problem.
- Simple procedures exist for controlling passband ripple in those special cases in which passband gain must be precisely controlled.
- Speed and storage needs of these filters appear to be competitive with the requirements of more conventional filter designs.

A brief review of the details of the interpolation operation will serve to introduce the filter synthesis work. The single level interpolator of Figure 2-3 creates two interleaved outputs. The original input signal (delayed) from the a_{-1} tap and the interpolation signal on the d_1 bus. If this processor is cascaded with a similar one as per Figure 3-1, the output of the second processor will contain, intact, the output of the first processor interleaved with new data generated by the second processor. The output of the second processor will, of course, also contain, intact, the input to the first processor as is obvious by induction. The data-rate doubling at each stage makes room for the old (input) to be interleaved with the new data generated at that stage.

In the general case of an L-level processor as per Figure 3-1 one can easily show that the delay through the processor in terms of output pulses is simply

$$D_i = \sum_{k=1}^{L} 2^{(L+1-k)} N_k - (2^L - 1) = N_f \qquad (4\text{-}1)$$

where N_k is the number of coefficients at level k (one-half the processor length). For $L > 5$, the linear interpolation operation of program C432 always inserts a leading zero into the impulse response of the equivalent interpolation filter. This produces a delay which is one greater than that given by Equation (4-1). This result is predicated on the use of $N_k = 1$ for $k > 5$ when the calculation is performed. The number of coefficients, N_f, on each side of the central (unity) term of the impulse response of the interpolation cascade is also given by the Equation (4-1) calculation. This will be discussed in more detail at a later point. The unit correction for D_i for $L > 5$ does not apply to the N_f calculation.

In terms of processor load it is convenient to define an "operation" as two additions plus one multiplication. Since $a_m = a_{-m}$ in these designs, one sums pairs of tap voltages, multiplies this sum by the coefficient, then sums to the d_1 bus. It is easy to show that the total number of interpolator operations per input sample is

$$O_i = \sum_{k=1}^{L} 2^{(k-1)} N_k \qquad (4\text{-}2)$$

Consider now a level one processor of the Figure 2-3 type operating in the context of Figure 3-1 with a γ_1, of, say, 80%. The spectrum of the level one output would appear as shown in Figure 3-2(b) where the sampling rate is 2R as compared to R at level one input. The spectrum of Figure 3-2(b) also represents the frequency function (Figure 2-4a) of the level one processor in the usual "Fourier transform of the impulse response" context, based on the output sampling rate 2R. Note that as a digital filter, the level one processor has a passband edge which is 40% of foldover or one-half the input design value.

The input sampling rate at each level becomes the foldover frequency at the output of that level. Thus, the second level filter sees an entire frequency-pattern period at its input transformed to fall within its foldover interval as defined at the output sampling rate. In Figure 3-2(c) an extended γ is used to place the entire spectral group centered at 2R in Figure 3-2(b) within the stopband of the second-level filter as previously discussed.

In Figure 3-2 the frequencies shown are absolute, but the γ relationships to foldover are referenced to input sampling rate values as was appropriate for the original interpolation discussions. For present purposes reference to output sampling rates is more appropriate, which involves simply a scaling factor of one-half. At the output of the third level, for example, the absolute passband edge frequency is still γ_1 (R/2) but the sampling rate here is 8R which means that the effective γ has been cut by a factor of eight. The passband edge at the output of level three is only 10% of foldover relative to the level three output sampling rate.

Comparing the original input signal spectrum of Figure 3-2(a) with the level three output spectrum of Figure 3-2(f) suggests a digital filter with a frequency function in the Figure 2-4a context with a $\gamma_1/8$ passband relative to foldover. The suggested low-pass filter function can be rationalized relative to the sampling rate change by assuming an 8R input sampling rate with seven zero samples following every actual sample. The extended γ choice at level two of Figure 3-2 insures that the original passband shape from level one is maintained essentially intact through an arbitrary number of interpolation levels. Thus in the frequency domain the net effect of each additional level is essentially one of scaling the passband portion of the frequency function down by a factor of two.

The sampling rate equalization artifice used above in order to make meaningful the frequency transfer function concept may be avoided completely. The impulse response of the cascade may be obtained by inputting a single unit value, followed by enough zero samples to force out the entire impulse response sequence. Once these numbers are available, a normal and equivalent filter structure may be constructed. Of course, if this equivalent filter is used for interpolation purposes computational advantage should be taken of the fact that only one in every 2^L input samples is nonzero. This topic will be touched upon again in the paper. The immediate concern, however, is the class of filters that may be derived from the impulse response of the cascade of Figure 3-1.

4.2 DESIGN EXAMPLES

Filters derived from the interpolator cascade, not surprisingly, have many of the same characteristics of the original interpolation process as discussed relative to Figure 2-2 (also see Tables 4-1 and 4-2). Computer program C432 in the Appendix contains 12-selectable cascade designs for interpolation identified by parameter NT. Designs for stopbands of -65, -50, -40, and -25 dB are available. The first-level N value selections for these designs are shown circled and numbered in Figure 2-2. Each cascade

TABLE 4-1

INTERPOLATOR/FILTER PARAMETERS

NT	Stopband Level (dB)	No. Charging Pulses, N_c* L = 1 - 5					Delay D_i and No. Filter Coefficients N_f (L = 1, 5)					No. Interpolator Operations O_i (L = 1, 5)				
		1	2	3	4	5	1	2	3	4	5	1	2	3	4	5
1	-65	20	25	27	27	28	19	47	99	201	405	10	20	32	48	80
2	-50	14	18	19	20	20	13	33	69	141	283	7	15	23	39	55
3	-40	10	13	14	15	15	9	23	49	99	199	5	11	19	27	43
4	-25	6	8	9	9	9	5	13	27	55	111	3	7	11	19	35
5	-65	38	43	45	45	46	37	83	171	345	693	19	29	41	57	89
6	-50	28	32	33	34	34	27	61	125	253	507	14	22	30	46	62
7	-40	20	23	24	25	25	19	43	89	179	359	10	16	24	32	48
8	-25	12	14	15	15	15	11	25	51	103	207	6	10	14	22	38
9	-65	10	13	14	15	15	9	23	49	101	203	5	11	19	35	51
10	-50	8	10	11	12	12	7	17	37	75	151	4	8	16	24	40
11	-40	6	8	9	9	9	5	13	27	55	111	3	7	11	19	35
12	-25	4	5	6	6	6	3	7	15	31	63	2	4	8	16	32

*For L > 5 use the L = 5 value divided by 2^{L-5} plus one.

TABLE 4-2

INTERPOLATOR TYPE DESCRIPTIONS

Interpolator Type (NT)	1st Level γ (%)	Stopband Level (dB)	Sampling Rate Type	γ Progression Level 2-5 (%)	N_k Values by Level 1-5
1	80	-65	Critical	60, 30, 15, 7.5	10, 5, 3, 2, 2
2	80	-50	Critical	60, 30, 15, 7.5	7, 4, 2, 2, 1
3	80	-40	Critical	60, 30, 15, 7.5	5, 3, 2, 1, 1
4	80	-25	Critical	60, 30, 15, 7.5	3, 2, 1, 1, 1
5	90	-65	Critical	60, 30, 15, 7.5	19, 5, 3, 2, 2
6	90	-50	Critical	60, 30, 15, 7.5	14, 4, 2, 2, 1
7	90	-40	Critical	60, 30, 15, 7.5	10, 3, 2, 1, 1
8	90	-25	Critical	60, 30, 15, 7.5	6, 2, 1, 1, 1
9	60	-65	Oversampled	30, 15, 7.5, 3.75	5, 3, 2, 2, 1
10	60	-50	Oversampled	30, 15, 7.5, 3.75	4, 2, 2, 1, 1
11	60	-40	Oversampled	30, 15, 7.5, 3.75	3, 2, 1, 1, 1
12	60	-25	Oversampled	30, 15, 7.5, 3.75	2, 1, 1, 1, 1

design was obtained by drawing horizontal lines on Figure 2-2 at the four stopband levels (taking into account the 6-dB gain provided by interleaving). Safe N values at the various cascade γ lines (see Table 4-2 for the γ-progresions) were selected for five levels and the appropriate coefficients for these interpolator designs were incorporated into the program.

As an example, the NT = 2 processor provides for -50 dB stopbands and starts with a first level N of 7 at a γ value of 80%. The second level γ is 60% with an N value of 4. The third, fourth, and fifth levels have N values of 2, 2, and 1, and γ values of 30, 15, and 7.5%. Designs NT = 1-8 use extended second-level γ values and are suitable for low-pass filter design. Designs NT = 9-12 are meant only for interpolation in the oversampled case. Their "stopband" usage mode for the derived filters will not be considered here.

If one considers use of the Figure 3-1 cascade for the derivation of impulse response data, it is interesting to follow the development of this response through the cascade. At each stage the input response is stretched and new values are interleaved into the vacated spaces as per Figure 2-3. The passband region of the frequency response function is essentially scaled down by two to one in the process. After a sufficient number of interpolator stages are traversed, continued scaling may be done by simple linear interpolation as previously discussed. Hence the frequency scaling operation can be extended easily in most cases beyond values of practical interest.

Examples of the above techniques are demonstrated in Figures 4-1 through 4-7. (All frequency functions in this paper are normalized to foldover frequency.) For this demonstration, design NT = 5 was chosen which is admittedly a "showboat" design. NT = 5 yields the best performance (transition region width is 22% of passband width, $W_T/f_p = 0.22$, sidelobe level = -65 dB), but also has the largest processor demand. The tests run were very simple: the impulse responses from the various levels of Figure 3-1 were individually extracted and written on disc files by the time-share program of the Appendix. At a later time a batch program read these files, performed on 8192-point Fast-Fourier transform (FFT) and plotted the results, normalized to the peak level found in the entire positive frequency space. The plots show the entire 4097-point spectrum with the foldover frequency referenced to unity at the far right of each plot.

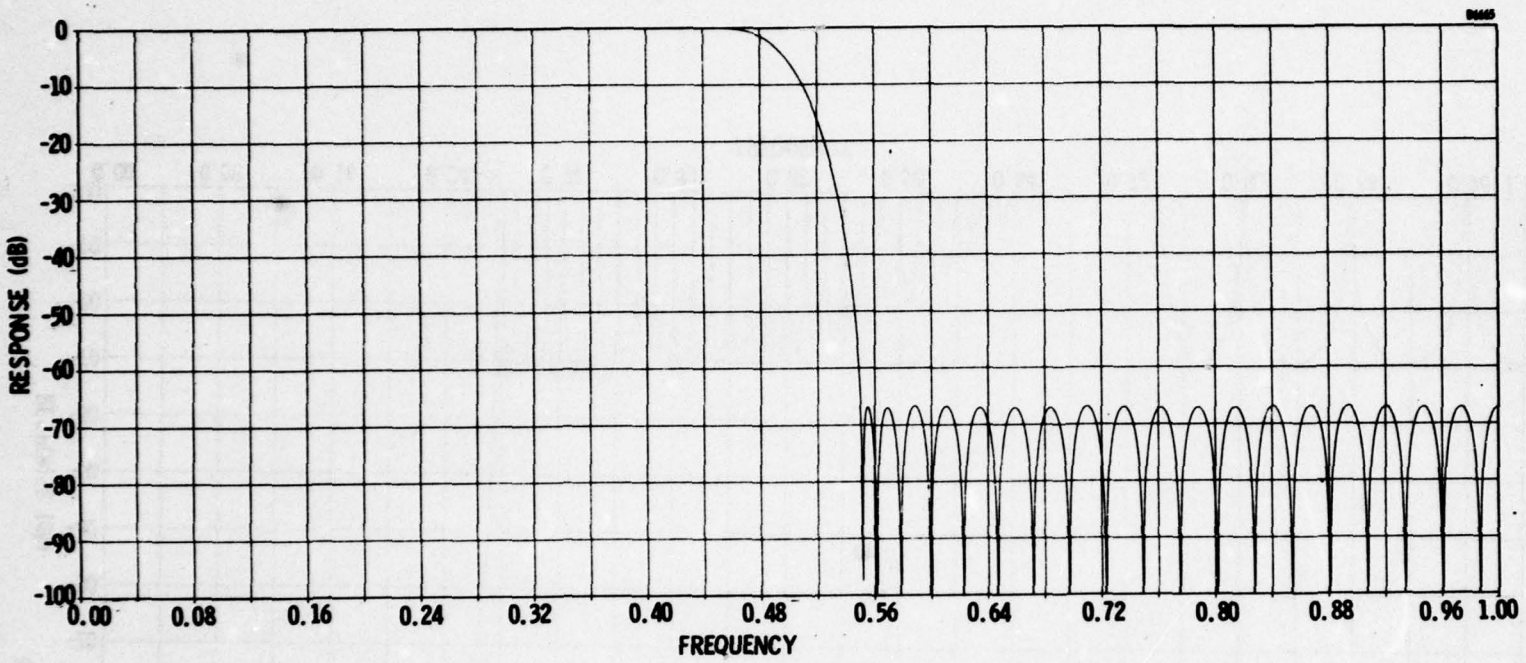


Figure 4-1. Filter Function, $\gamma_1 = 90\%$, NT = 5, L = 1, $f_p = 0.45$, $N_f = 37$, Peak Passband Ripple = -67.09 dB

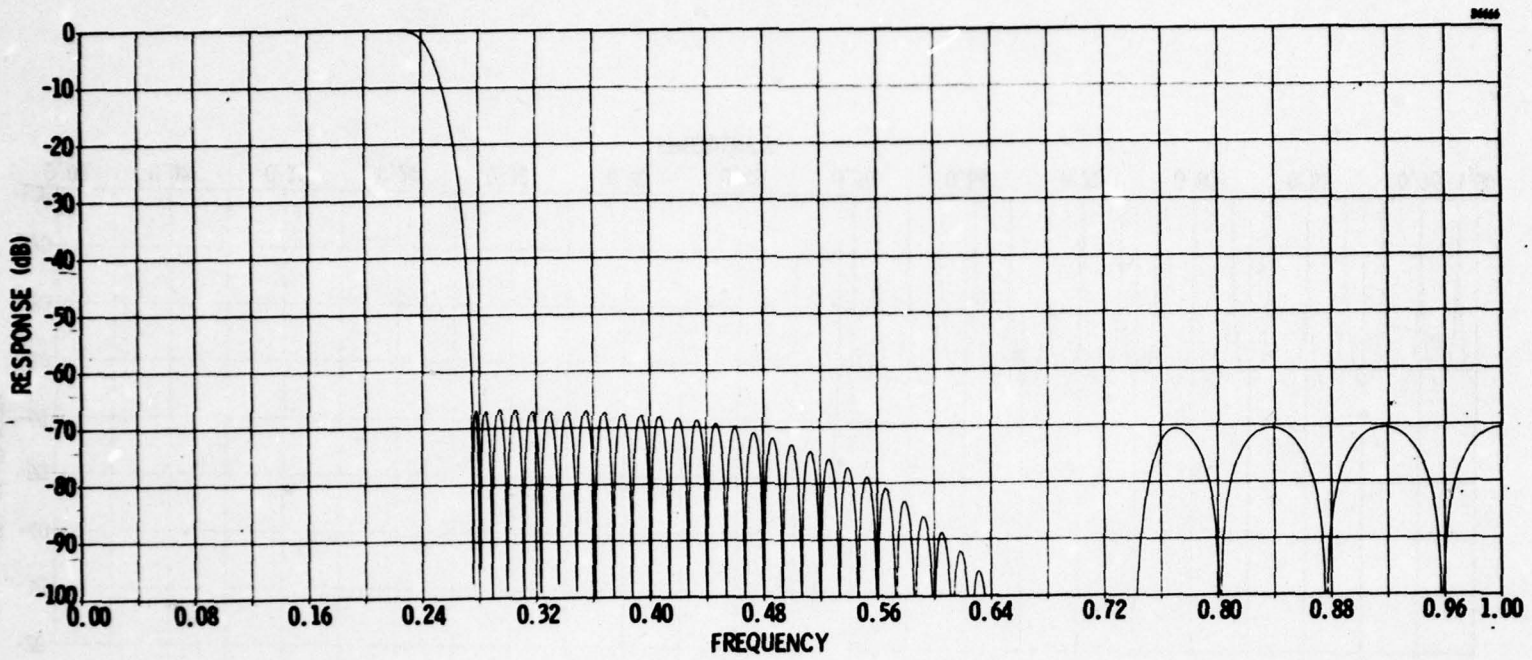


Figure 4-2. Filter Function, γ_1 = 90%, NT = 5, L = 2, f_p = 0.225, N_f = 83, Peak Passband Ripple = -62.63 dB

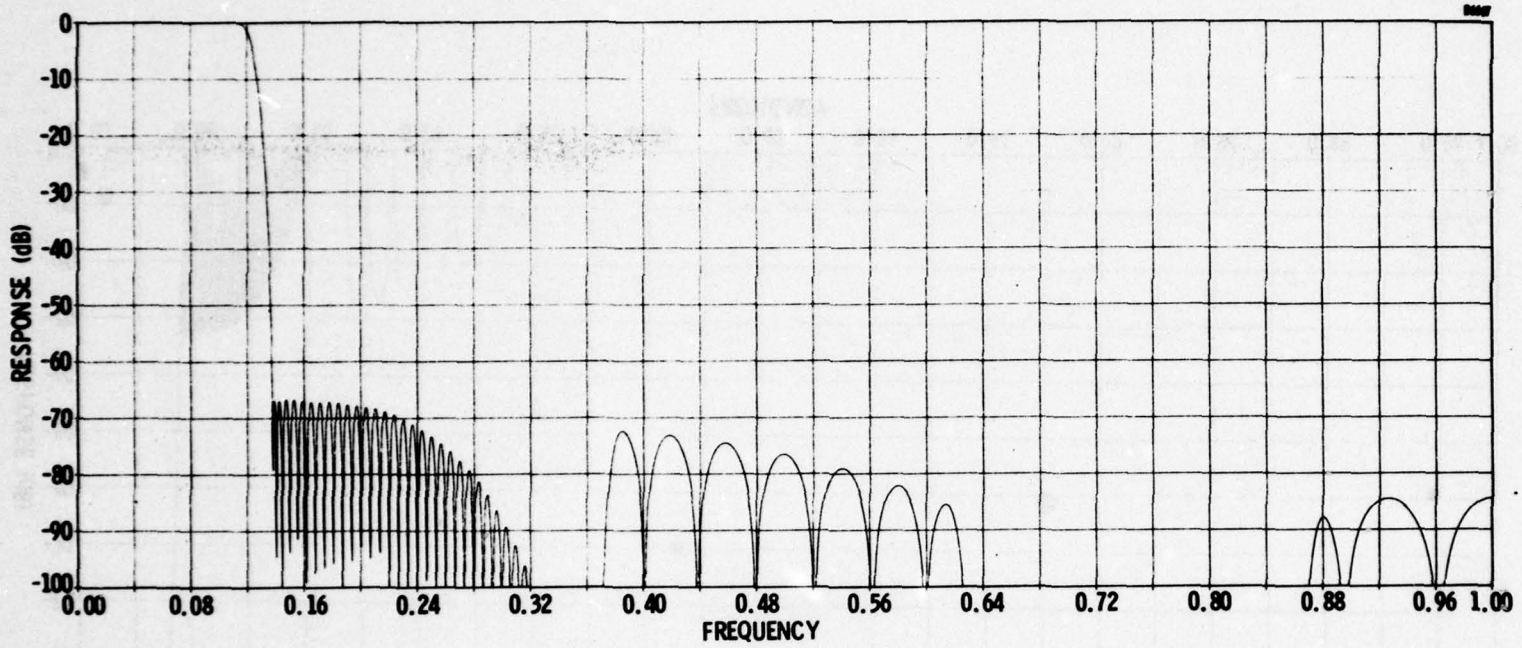


Figure 4-3. Filter Function, $\gamma_1 = 90\%$, NT = 5, L = 3, $f_p = 0.1125$, $N_f = 171$, Peak Passband Ripple = -61.95 dB

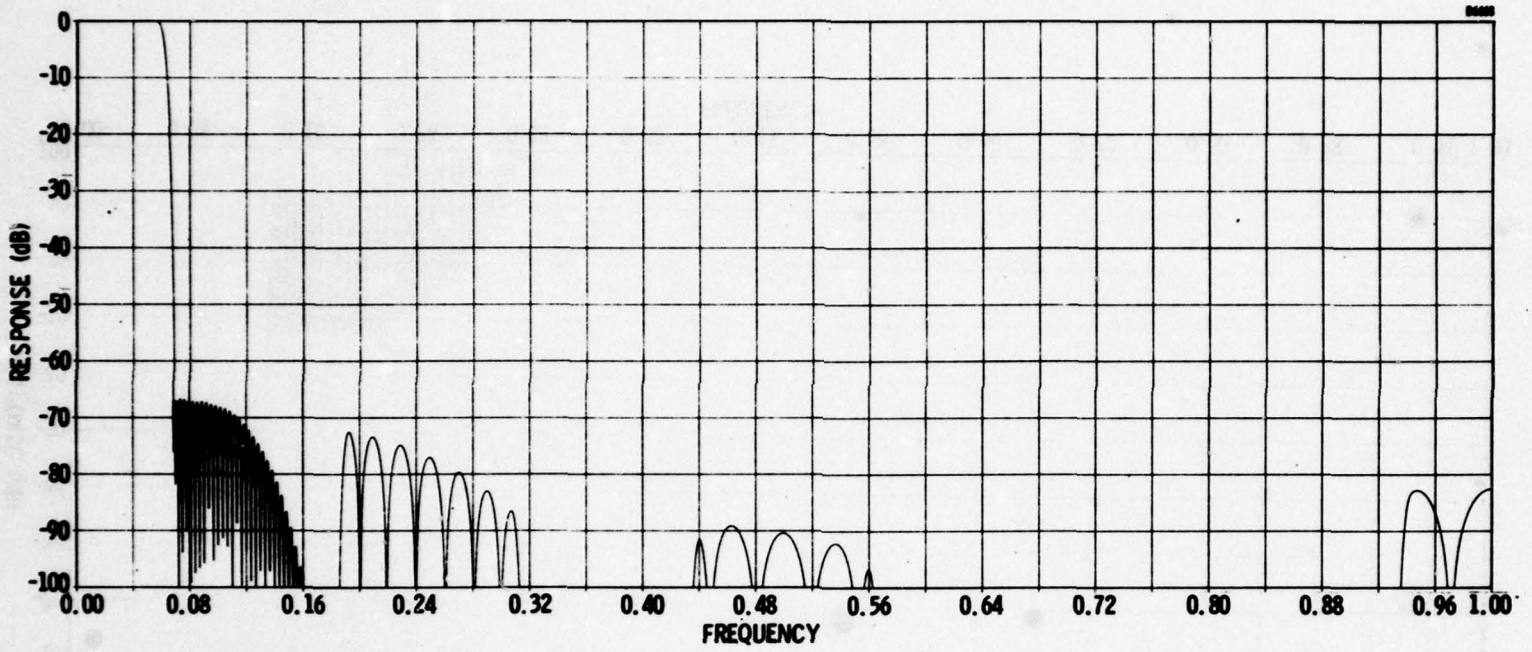


Figure 4-4. Filter Function, γ_1 = 90%, NT = 5, L = 4, f_p = 0.05625, N_f = 345, Peak Passband Ripple = -62.80 dB

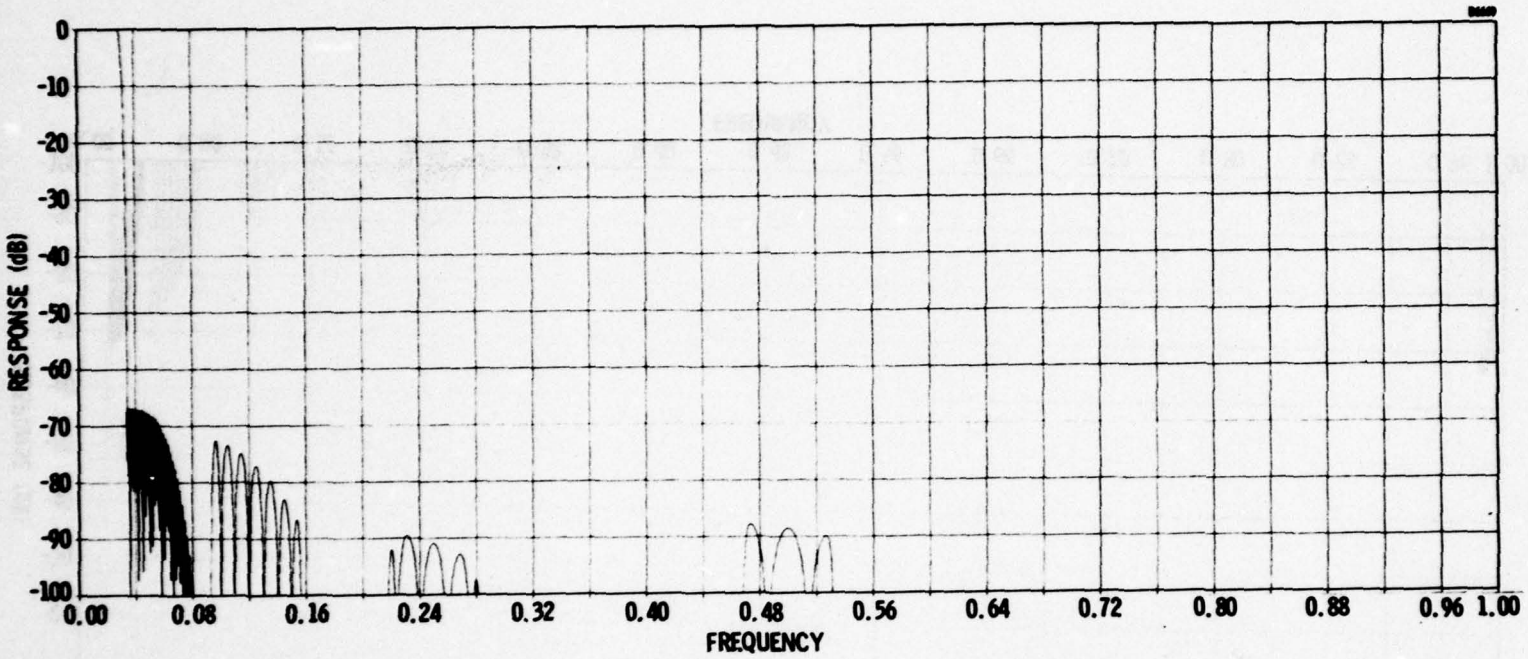


Figure 4-5. Filter Function, γ_1 = 90%, NT = 5, L = 5, f_p = 0.028125, N_f = 693, Peak Passband Ripple = -62.80 dB

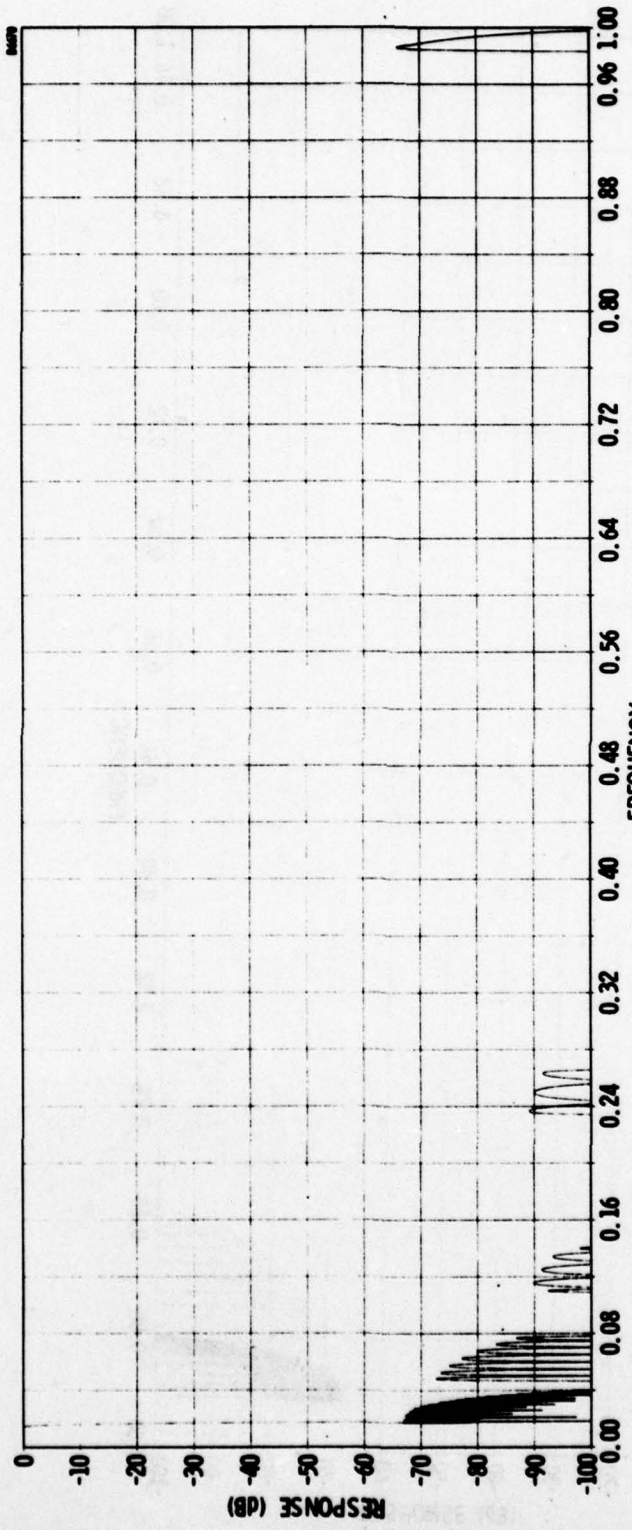


Figure 4-6. Filter Function, $\gamma_1 = 90\%$, NT = 5, L = 6, $f_p = 0.0140625$, $N_f = 1387$, Peak Passband Ripple = -59.08 dB

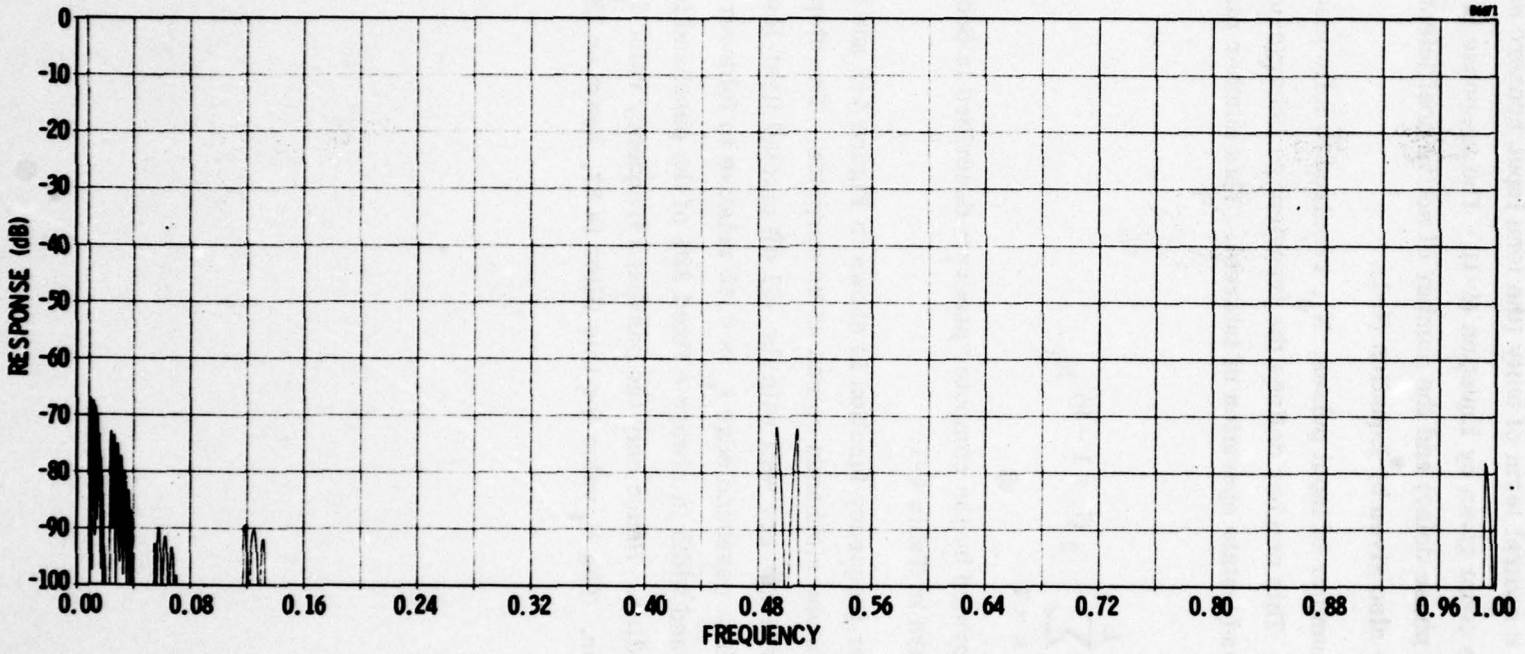


Figure 4-7. Filter Function, γ_1 = 90%, NT = 5, L = 7, f_p = 0.00703125, N_f = 2775, Peak Passband Ripple = -61.26 dB

The impulse responses which result (see par. 4.5) from this synthesis procedure and computer program have a central term of unity (the lone input nonzero sample) which is delayed in the file by the count given by Equation (4-1). The response is symmetric about the central term (linear phase delay) and the number of active coefficients on each side of the central term, N_f, is also given by Equation (4-1).

Knowledge of the number of input pulses, N_c, required to fully load the interpolation cascade is often useful. This number defines the transient or "charge-up" period which precedes the normal steady-state operation of interest. This number may be calculated from

$$N_c = \frac{1}{2^{L-1}} \sum_{k=1}^{L} 2^{(L+1-k)} N_k . \qquad (4-3)$$

This calculation is performed by the computer program described in Section 6, par. 6.4 and tabulated data appears in Table 4-1.

The level one filter frequency function is shown in Figure 4-1 and the Chebyshev behavior in both the pass and stopbands comes as no surprise. The stopband is at about the -67 dB level which checks very well with the -61 dB circled first-level choice for NT = 5 in Figure 2-2. The passband edge f_p is 0.45 relative to foldover (one-half of 0.90). The transition band width is always a fixed 22% of the passband in these figures due to the frequency scaling. Hence only the passband frequency value f_p need be given to define the filter behavior. The N_f value for this filter is 37, however 18 of these coefficients are zero.

Figure 4-2 shows the level two filter function with $f_p = 0.225$. The level one sidelobes are compressed in frequency due to the stretching in time. If there were no level two fill of the vacated spaces, the level one passband would appear at the far right of Figure 4-2. The net effect of the level two fill in the frequency domain is very nearly a multiplication of the level one passband with level two stopband. Thus the -70 dB sidelobes at the right of the figure are the coarse sidelobes of the level two interpolator which has an N value of 5 (compared to 19 for the first level). The -70 dB sidelobes agree very well with the -64 dB, N = 5, γ = 60% value from Figure 2-2.

Note also in Figure 4-2 that the level one sidelobes are being affected by the level two transition region which extends from 0.3 to 0.7 and is 6-dB down at 0.5. As additional levels are traversed, the transition region effects cumulate to drive much of the sidelobe structure below -100 dB as will be seen. The N_f for this filter is 83.

The "stretch-and-fill" pattern effects of Figure 4-3 for the level three response show a stopband at about -84 dB. This again agrees well with the Figure 2-2, N = 3 value at γ = 30% of -78 dB. The transition region for level three extends from 0.15 to 0.85. The passband frequency f_p for this filter is 0.1125 and the N_f is 171.

The four-level equivalent filter of Figure 4-4 has an f_p of 0.05625 and an N_f of 345. The level four stopband at -82 dB corresponds to a -76.5 dB level at N = 2 for γ = 15% in Figure 2-2. The transition region of level four extends from 0.075 to 0.925.

The level five response of Figure 4-5 has an $f_p = 0.028125$ and an N_f of 693. No response is seen near the foldover frequency since an N of 2 at γ = 7.5% (level 5 design) places the stopband response near -107 dB and hence out of sight. The transition region here extends from 0.0375 to 0.9625.

The level six performance shown in Figure 4-6 is especially interesting as this represents a departure in operating procedures. The interpolation program provides Figure 2-3 type processors at each of the first five levels. After the fifth level simple linear interpolation is provided. The -59 dB control line (for -65 dB performance) drawn in Figure 2-2 which was followed for levels 2, 3, 4, 5 at γ values of 60, 30, 15, 7.5% would indicate one more stage of N = 1 at 3.75% for the sixth level. The optimum N = 1 designs have 6-dB less peak error than straight interpolation as was discussed relative to Equations (2-13) through (2-16); linear interpolation is equivalent to $b_1 = 1$ in Equation (2-17). The N = 1 point of Figure 2-1 for γ = 3.75% indicates that linear interpolation at the sixth level would provide a stopband level of -62 dB. (The 6-dB gain of the note of Figure 2-1 is exactly offset by the 6-dB loss of linear interpolation.)

The actual peak response level noted near the foldover frequency of Figure 4-6 is closer to -66 dB which is within the -65 dB performance specification. The restricted response width out of level five ($f_p = 0.028125$) is the saving factor. This f_p value, when substituted into Equation (2-13) for $f/(R/2)$ with $b_1 = 1$, reveals an error level of -60.21 dB at band edge and when corrected by the 6-dB gain from interleaving, gives a -66.21 dB response level which confirms the Figure 4-6 results. Note also the sharpness of the response drop on the left side. Comparison of measurements from 0 to passband edge and from 1 to the sharp drop in the null region response also confirm the causal mechanism cited. At this level f_p is 0.0140625 and N_f is 1387. With linear interpolation or N = 1 one might consider that the entire frequency space is a transition region.

Figure 4-7 shows level seven output which is the result of five levels of 1:2 interpolation and one level of 1:4 linear interpolation. A calculation similar to the one for level six predicts a stopband peak of -78.25 dB which is confirmed by the Figure. The f_p here is 0.00703125 and the N_f is 2775. It is clear that this scaling process could be continued further without difficulty.

The three filter functions in the figures which follow were all taken at level five in the $\gamma = 90\%$ set so that $f_p = 0.028125$ and $W_T/f_p = 0.22$ for all of these. Figure 4-8 shows the NT = 8 design which was specified for -25 dB stopbands and used an N_k sequence of 6, 2, 1, 1, 1. This high stopband level keeps the entire stopband region in view. The interpolator levels responsible for sidelobe groups seen in left-to-right order are: 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1, 5.

For this high sidelobe design the mainlobe of passband ripple is large enough to be visible in the plot. Calculations made with an auxiliary test program indicate that the peak passband ripple deviation is at the level of -22 dB. This compares with a peak ripple level of -26.7 dB at the output of level one. The peak passband ripple level increased by 4.7 dB in transit through the four levels two through five.

Figure 4-9 is next in this series with an NT value of 7 which is specified at -40 dB sidelobes. The N_k sequence here is 10, 3, 2, 1, 1.

Finally, for this group Figure 4-10 for NT = 6 gives results for the -50 dB design which used an N_k sequence of 14, 4, 2, 2, 1. The results are consistent with expectations.

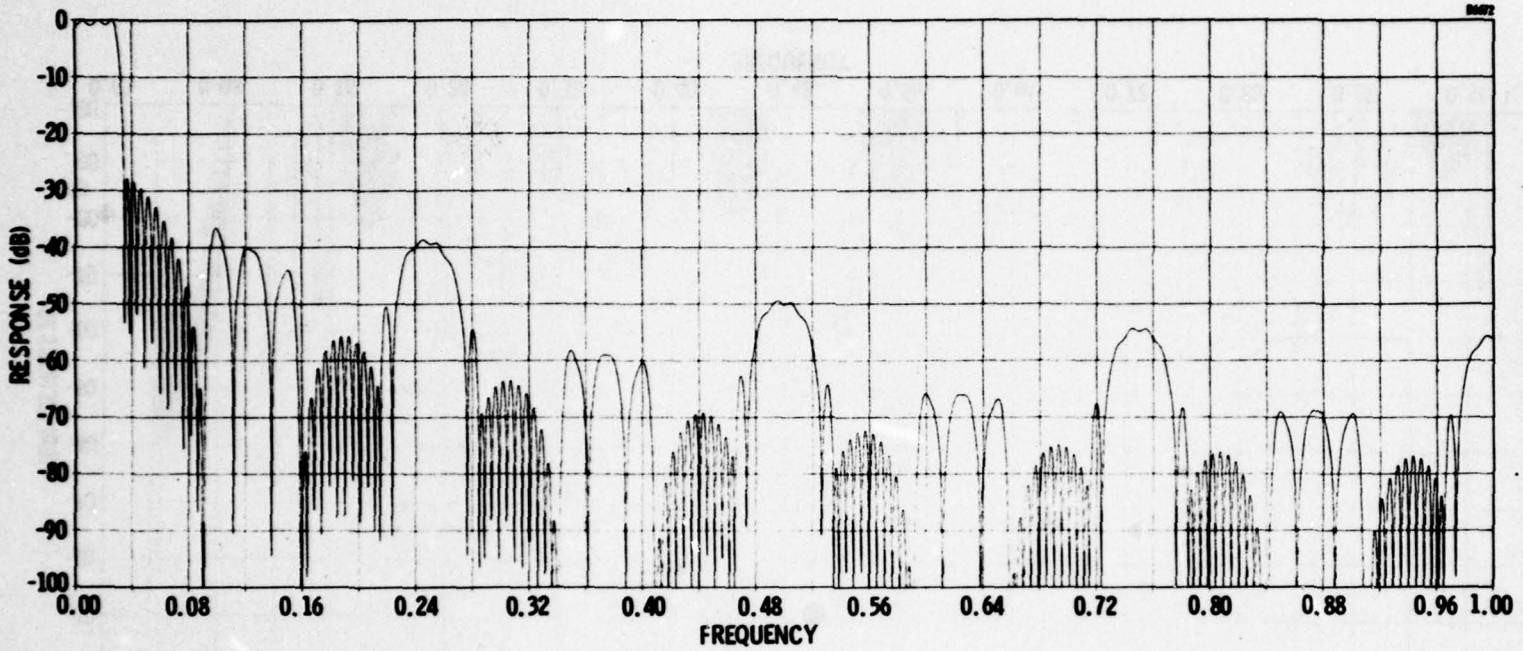


Figure 4-8. Filter Function, γ_1 = 90%, NT = 8, L = 5, f_p = 0.028125, N_f = 238, Peak Passband Ripple = -22.04 dB

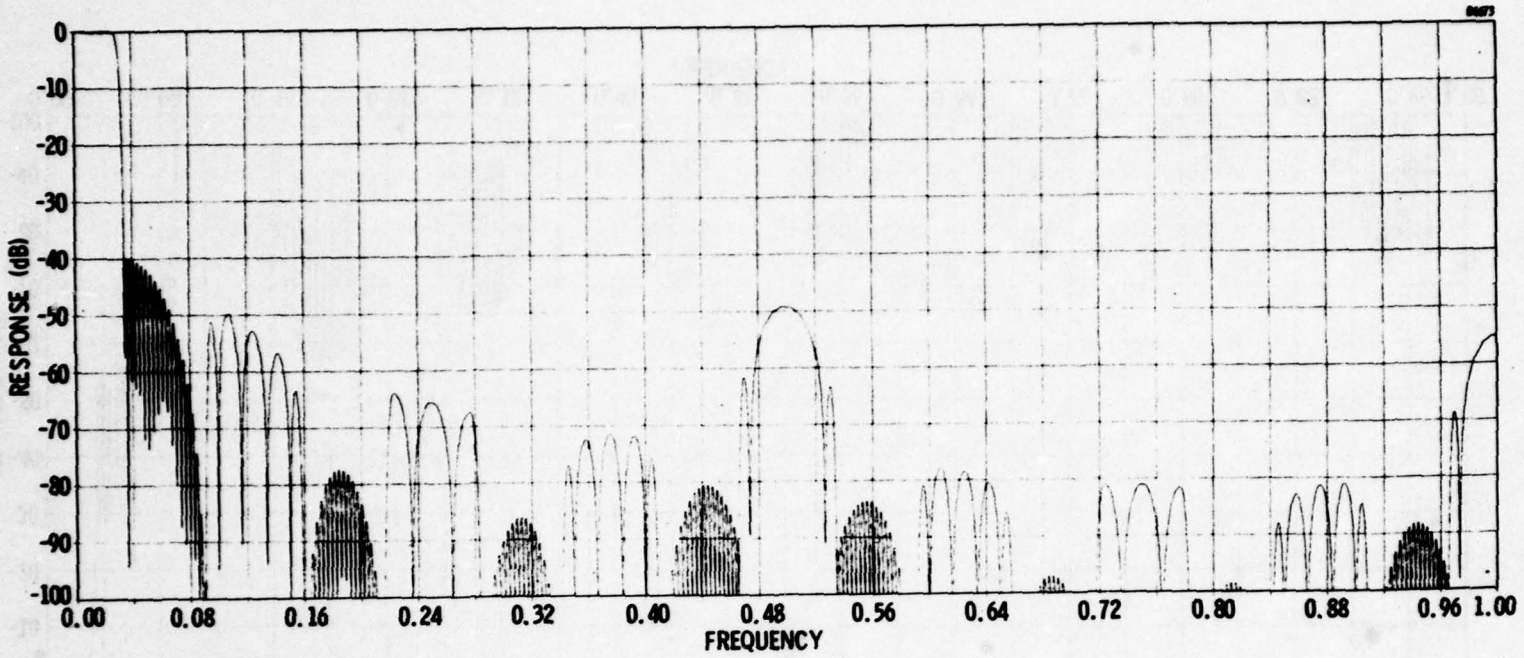


Figure 4-9. Filter Function, γ_1 = 90%, NT = 7, L = 5, f_p = 0.028125, N_f = 390, Peak Passband Ripple = -33.09 dB

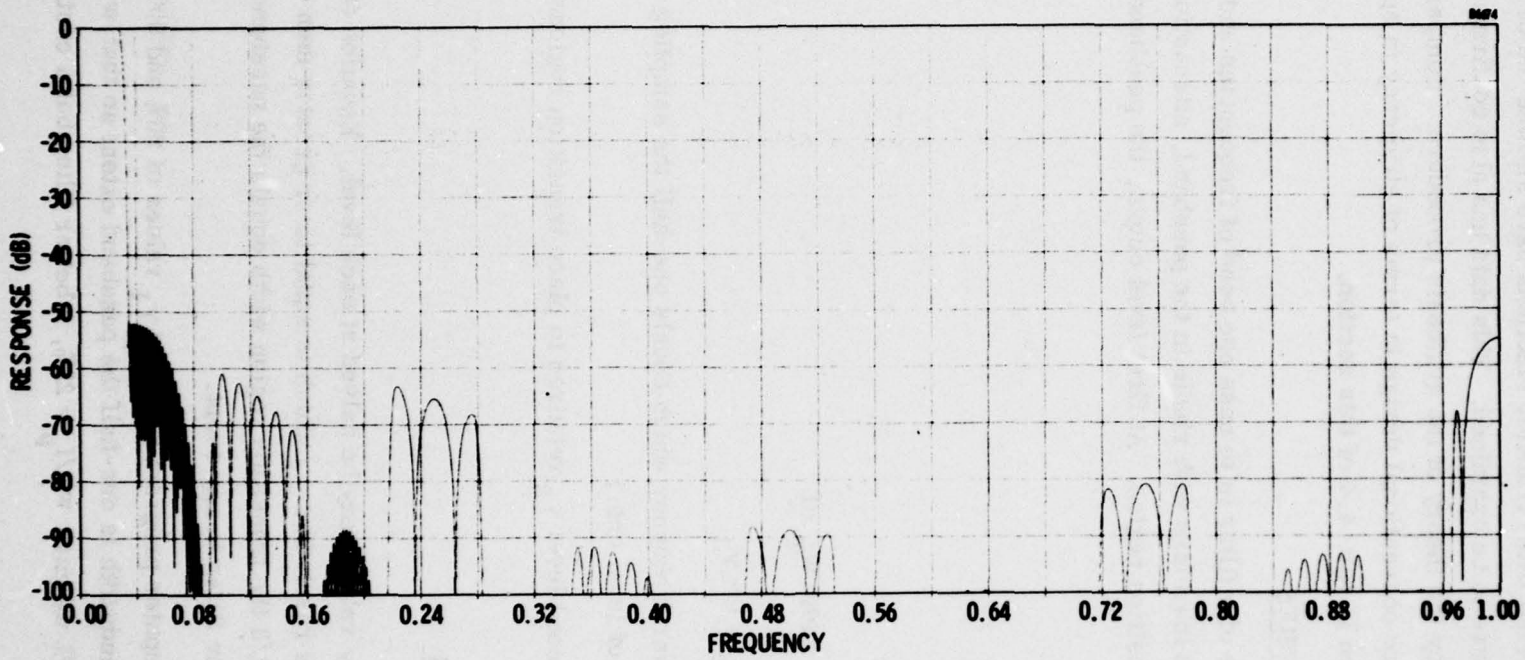


Figure 4-10. Filter Function, $\gamma_1 = 90\%$, NT = 6, L = 5, $f_p = 0.028125$, $N_f = 538$, Peak Passband Ripple = -46.25 dB

These plots of derived filter transfer functions have allowed the detailed workings of this synthesis procedure to be examined. This data has also confirmed quantitatively the performance predicted by the theory of the synthesis procedure. Comparisons of these filters with those of more conventional design in terms of efficiency in application and other similar factors are given in par. 4.4 of this section.

4.3 DESIGN PROCEDURES

The basic purpose of a filter is to pass one band of frequencies and exclude another band. One is concerned therefore with ripple in the passband, attenuation in the stopband and the width of the transition region. At first level output, the passband edge f_p is

$$f_p = \frac{\gamma_1}{2} \tag{4-4}$$

The stopband edge f_s is

$$f_s = 1 - \frac{\gamma_1}{2} \tag{4-5}$$

with a transition region width w_T of

$$w_T = f_s - f_p = 1 - \gamma_1 \tag{4-6}$$

all relative to the foldover frequency which equals one-half the sampling rate. (Sampling rate is 2R at the output of level one.)

The overlap in second level γ_2 required to place transition regions in the stopband requires that

$$\gamma_2 \geq 1 - \frac{\gamma_1}{2} \tag{4-7}$$

From this level on the γ values may be halved at each level. Equation (4-7) shows that γ_1 values of 2/3 or less result in a γ_2 which is equal to or greater than γ_1 which increases filter length. At $\gamma_1 = 2/3$ the transition region width equals the passband bandwidth. This represents a rather poor filter in any event.

The Appendix computer program provides γ_1 values of 80% and 90% for filtering. At 80% the transition bandwidth is one-half the passband extent so that $w_T/f_p = 50\%$ for $\gamma_1 = 80\%$. For $\gamma_1 = 90\%$ we have $w_T/f_p = 22\%$, a better filter but a costlier one.

The design procedure is quite simple:

1. Select transition region relative width $\gamma_1 = 80\%$ (NT = 1 - 4) or $\gamma_1 = 90\%$ (NT = 5 - 8), currently available.
2. Select stopband level (-25, -40, -50, -65 dB), NT is now identified. (See Tables 4-2 and 4-2.)
3. Normalize desired filter cutoff frequency f_c by use of Equation (4-13) to obtain f_p'. Compute frequency compression ratio C_f

$$C_f = \gamma_1 / f_p' \tag{4-8}$$

4. If C_f is acceptably close to a power of 2, compute

$$L = LEV = \log_2 (C_f) \tag{4-9}$$

(if this is not possible go to step 6.)

5. Use the Appendix program (XLINT, LINT) to generate the desired filter impulse response. Task complete.
6. If the C_f factor cannot be approximated sufficiently well by a power of 2, find and L and J such that

$$C_f \cong \frac{2^L}{J} \tag{4-10}$$

(See Section 5 par. 5.4 for details of L, J determination.)

(a) Using LEV = L get impulse response data as in step 5.

(b) Derive desired symmetric filter impulse response by using central term (unity) of (a) above and selecting every J^{th} term thereafter. Task complete.

For an example of a "J-derived" filter consider the following: Let sharpness of cutoff require $\gamma_1 = 90\%$. Let sidelobe level specification be < -60 dB, so NT = 5 is selected. Assume clock period in processor hardware of 200 μs or 5000/s repetition rate, and assume filter cutoff frequency $f_c = 225$ Hz ± 2%. Normalize this cutoff frequency to the foldover frequency as per Equation (4-13).

$$f_p' = \frac{f_c}{\frac{R}{2}} = \frac{225}{\frac{5000}{2}} = 0.09$$

then from Equation (4-8)

$$C_f = \gamma_1/f_p' = 0.9/0.09 = 10$$

This is not a power of two, go to step 6 and try

$$C_f = 10 \tilde{=} \frac{2^7}{13} = 9.8462$$

This will give f_c = 228.52 Hz which is within the ±2% tolerance.

Use LEV = 7, NT = 5 and get impulse response from Appendix program C432. From Equation (4-1) we find that the central peak is delayed by

Delay = 2775 + 1 = 2776.

(The central peak is in cell number 2777 of the array.) Equation (4-1) states that there are

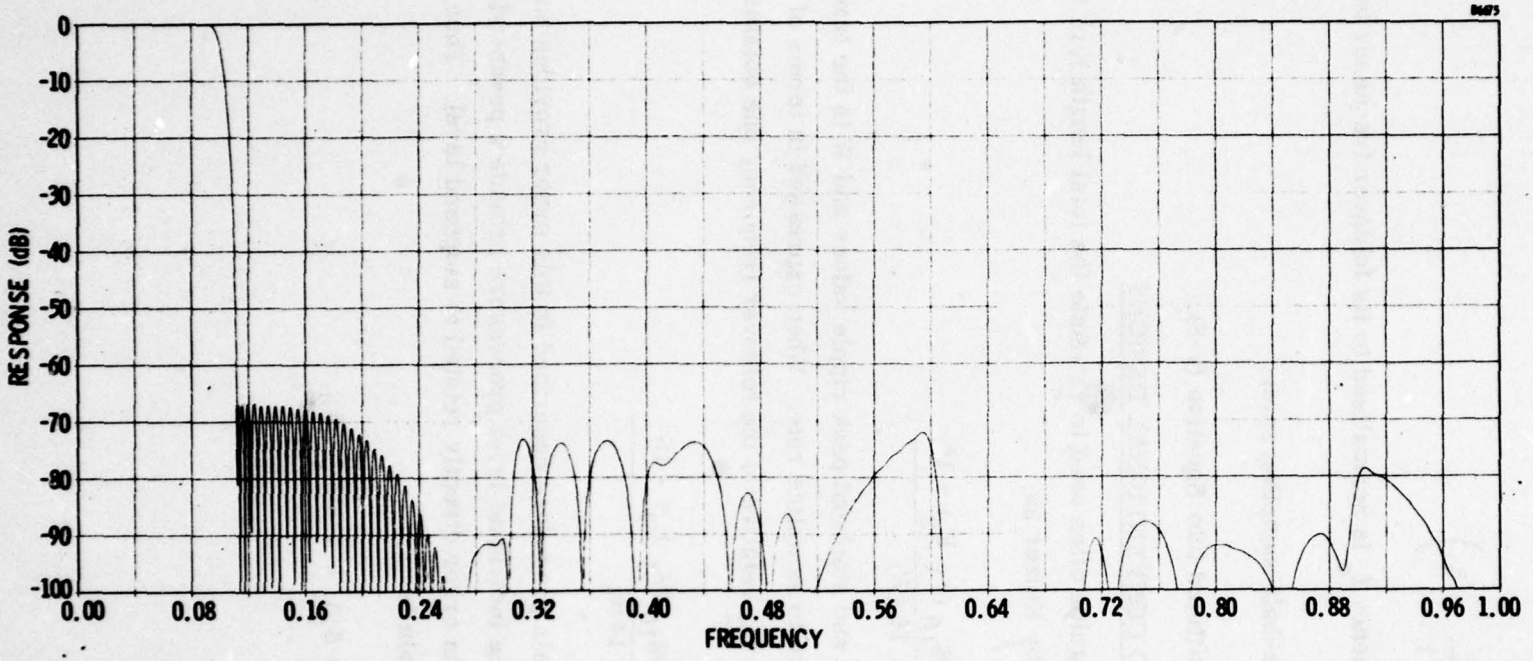
$$N_f = 2775$$

active coefficients on each side of unity. After every 13th element is selected and the new impulse response is created, it will have a delay of 213 and an N_f = 213. Figure 4-11 shows the frequency function for this filter. The impulse response involved in Figure 4-7 was *censored to yield the impulse response* of the Figure 4-11 data.

Figure 4-11 could also be compared to Figures 4-3 and 4-4 which are the closest "normal-sequence" filters. Apparently no real violence was done to the basic frequency response by the resampling process. This suggests that very-high LEV impulse responses be calculated and stored for a particular combination of transition width/passband ratios and sidelobe levels. These stored responses could then simply be resampled by some appropriate factor J for specific cutoff frequencies.

In this regard the three equations which follow may be helpful in defining the filter. Once the transition bandwidth to passband frequency ratio (w_T/f_p) is known, the first-level γ becomes

$$\gamma_1 = \frac{1}{\frac{w_T}{2f_p} + 1} \qquad (4\text{-}11)$$



NOTE

NT = 5, L = 7, DATA RESAMPLED 1:13. $W_T/f_p = 0.22$, $f_p = 0.0014$, $N_f = 213$, PEAK PASSBAND RIPPLE = −62.81 dB. SEE FIGURE 4-20 FOR IMPULSE RESPONSE.

Figure 4-11. Filter Function for "J-Derived" Filter

or

$$\frac{w_T}{f_p} = 2\left(\frac{1}{\gamma_1} - 1\right) \tag{4-12}$$

A specified cutoff frequency f_c is normalized to its foldover frequency to yield f_p'

$$f_p' = f_c/(\text{one-half sampling rate}) \tag{4-13}$$

which may then be substituted into Equation (4-8).

4.4 COMPARISON TO CONVENTIONAL DESIGNS

A rule-of-thumb guide often used to estimate the total length K of a low-pass digital filter is (Ref. 4) given by Kaiser as

$$K = \frac{-10 \log_{10}(E_1 E_2) - 15}{14 (W)} \tag{4-14}$$

where E_1, E_2 are pass and stopband peak ripple values and W is the transition region width expressed relative to sampling rate. When expressed in terms of filter half length N_f and transition width w_T relative to the foldover frequency one obtains

$$N_f = \frac{-10 \log_{10}(E_1 E_2) - 15}{14 w_T} \tag{4-15}$$

The filter synthesis procedure described in this paper provides no direct control of passband ripple. The individual level processors provide a passband gain proportional to $(1 + \delta)$ where δ is an error directly related to stopband level. Thus, through two stages, the passband gain is

$$(1 + \delta_1)\ (1 + \delta_2) \cong 1 + \delta_1 + \delta_2 \tag{4-16}$$

All stages have nearly the same δ value and since the stages have different γ values, it seems reasonable to assume that the δ's will add on an rms basis. A fair estimate of ripple might then be obtained and substituted into Equation (4-15) so that equivalent standard-design filter sizes might be estimated. This yields

$$N_f \tilde{=} \frac{-10 \log_{10}\left(\sqrt{L}\ \delta^2\right) - 15}{14\ w_T} \tag{4-17}$$

or

$$N_f \tilde{=} \frac{-20 \log_{10}(\delta) - 5 \log_{10}(L) - 15}{14\ w_T} \tag{4-18}$$

The first numerator term above is the negative of the stopband level in dB.

The filters described in this paper may be sized assuming that w_T/f_p is given so that γ_1 may be computed from Equation (4-11). Convenience is gained with no loss of generality if comparisons are normalized to level one output conditions. This results in a w_T value of $(1 - \gamma_1)$ according to Equation (4-6). The w_T at level L, w_{TL} would then be

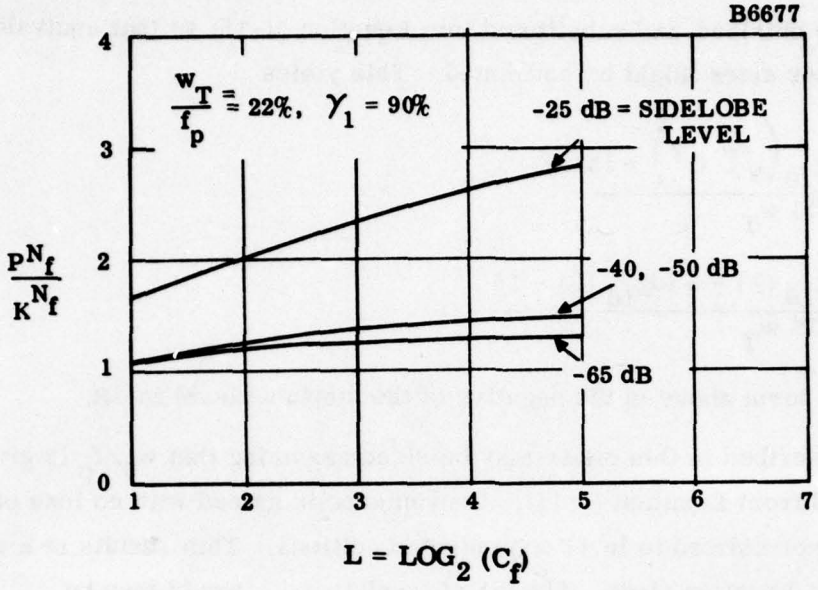
$$w_{TL} = \frac{(1 - \gamma_1)}{2^{L-1}} \tag{4-19}$$

This allows Equation (4-18) to be written as

$$N_f = \left[\frac{-20 \log_{10}(\delta) - 15 - 5 \log_{10}(L)}{14\ (1 - \gamma_1)}\right] 2^{L-1} \tag{4-20}$$

Once the L value is selected, Equation (4-1) or Table 4-1 may be used to obtain the required N_f which may be compared to that obtained from Equation (4-20).

Letting ${}_PN_f$ be the coefficient count of the filter design of this paper and ${}_KN_f$ be the corresponding Kaiser (Ref. 4) estimate, the results of Figure 4-12 obtain. The paper design appears to require more coefficients, but the ratios are reasonable. The passband ripple problem of the paper procedure shows to its disadvantage at the lower stopband suppression levels, as expected. For the higher-quality filters the paper design compares reasonable well with more conventional synthesis procedures. (Later passband ripple studies indicate that the ratios of Figure 4-12 are too large for the higher L values).



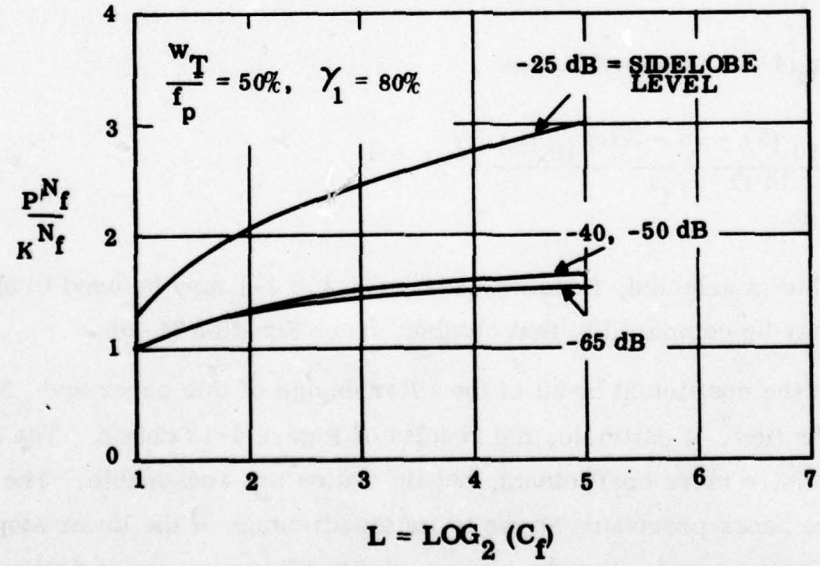


Figure 4-12. Filter Length Comparison

4.5 TIME DOMAIN CONSIDERATIONS

In previous discussions the filter frequency function was examined at different levels in Figure 3-1 structure. A similar study will be made here of the filter impulse response function. The case NT = 5 interpolator places serious restrictions on stopband level and the processor is nontrivial at all levels (N_k = 19, 5, 3, 2, 2). Figure 4-13 shows the first level impulse response which has an N_f = 37. (The sample values are the significant items of concern, the CalComp plotter draws straight lines between points. The discontinuities in slope indicate point location.) The frequency function for this data appears in Figure 4-1.

The second level processor is also a sophisticated one (N_2 = 5) so that when the first level input values are interleaved with new derived values the output of Figure 4-14 results. Considerable smoothing has been done but slope breaks are still evident. The N_f value is 83. The corresponding frequency function is that of Figure 4-2.

The third level processor has an N_3 = 3 and its "stretch-and-fill" action produces the considerably smoothed data of Figure 4-15 (frequency function is Figure 4-3). Slope granularity is still in evidence near local peaks. The N_f for this data is 171.

Fourth level output in Figure 4-16 is very smooth; the processor sophistication has dropped to N_4 = 2 here. N_f is 345 and the corresponding frequency function is that of Figure 4-4.

Another N_5 = 2 stage at level five produces the output shown in Figure 4-17 which has an N_f = 693 and corresponds to the frequency function of Figure 4-5.

Beyond this point in the chain, linear interpolation may be used. The "J-derived" filter of Figure 4-11 has the impulse response shown in Figure 4-18 where N_f = 213. This filter was derived by using L = 7 and then selecting every 13th sample of the resulting impulse response.

This rather attractive smoothing of the impulse response with an increase in L does not always occur. Consider Figures 4-19 and 4-20. Here NT = 8 (γ_1 = 90%, -25 dB stopband) and the $\{N_k\}$ sequence is 6, 2, 1, 1, 1. Because of the high sidelobe levels, the stages of Figure 3-1 beyond the second are just modified linear interpolators (N_k = 1). The level one output is coarse as expected (N_f = 11). However note that the L = 7 output of Figure 4-20 shows considerable slope granularity. The high sidelobe specification (-25 dB) permits this kind of crude "stretch-and-fill" operation. (The Y-axis scaling for all impulse response plots is the same. There are subtle differences in the X-axis positioning and scaling for the different plots.)

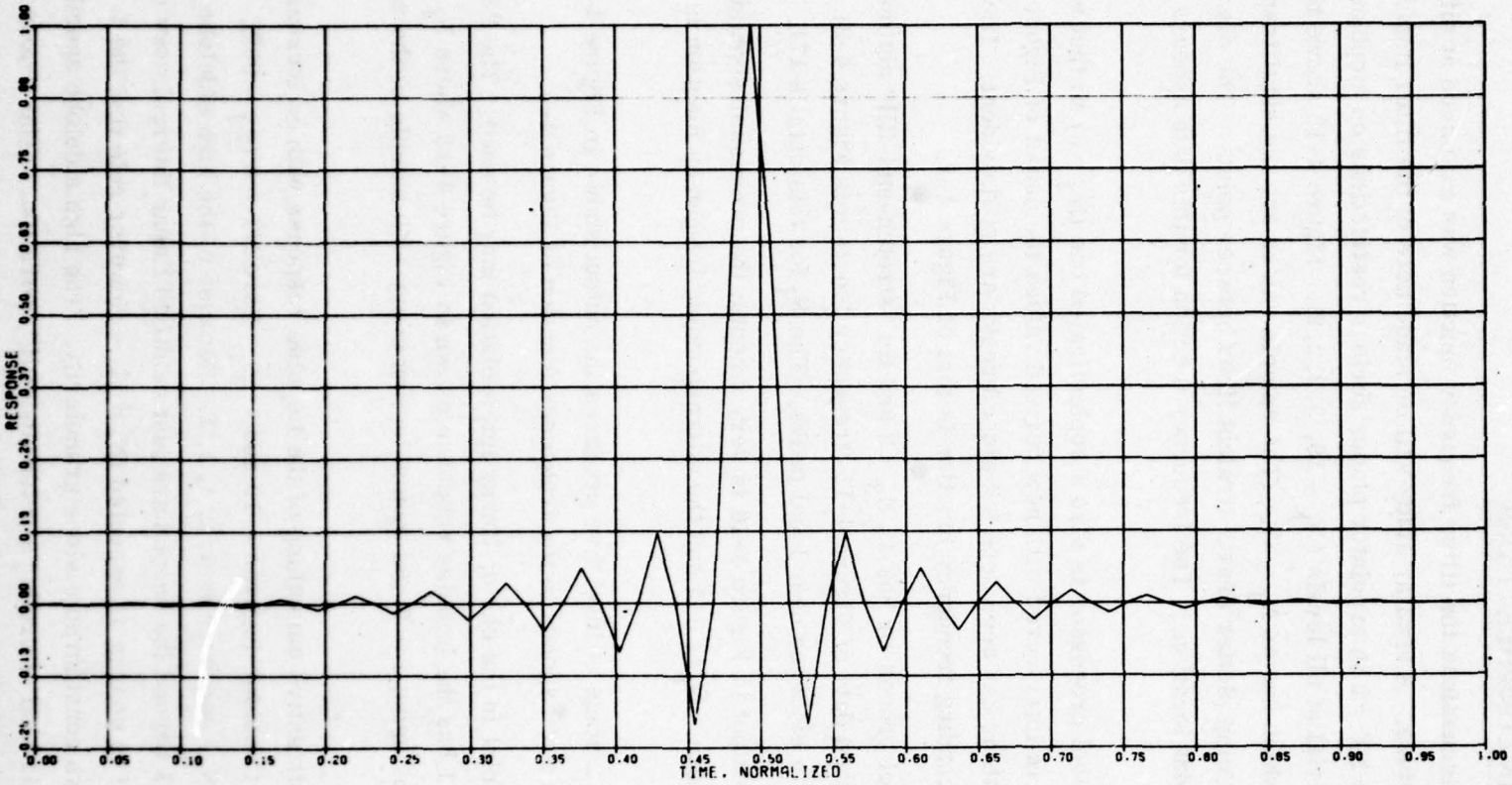


Figure 4-13. Filter Impulse Response, NT = 5, LEV = 1, N_f = 37, Frequency Function is Figure 4-1

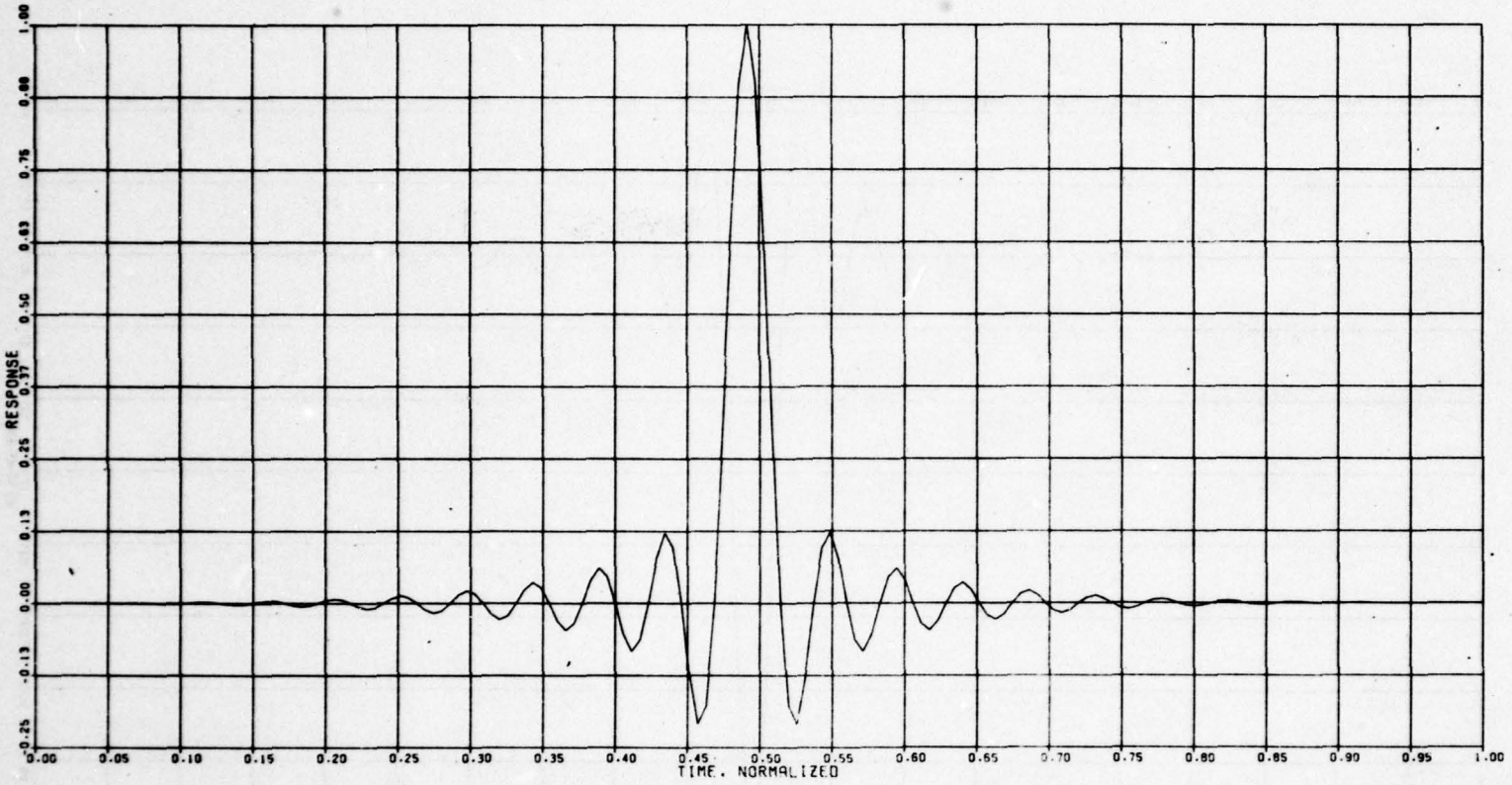


Figure 4-14. Filter Impulse Response, NT = 5, LEV = 2, N_f = 83. Frequency Function is Figure 4-2.

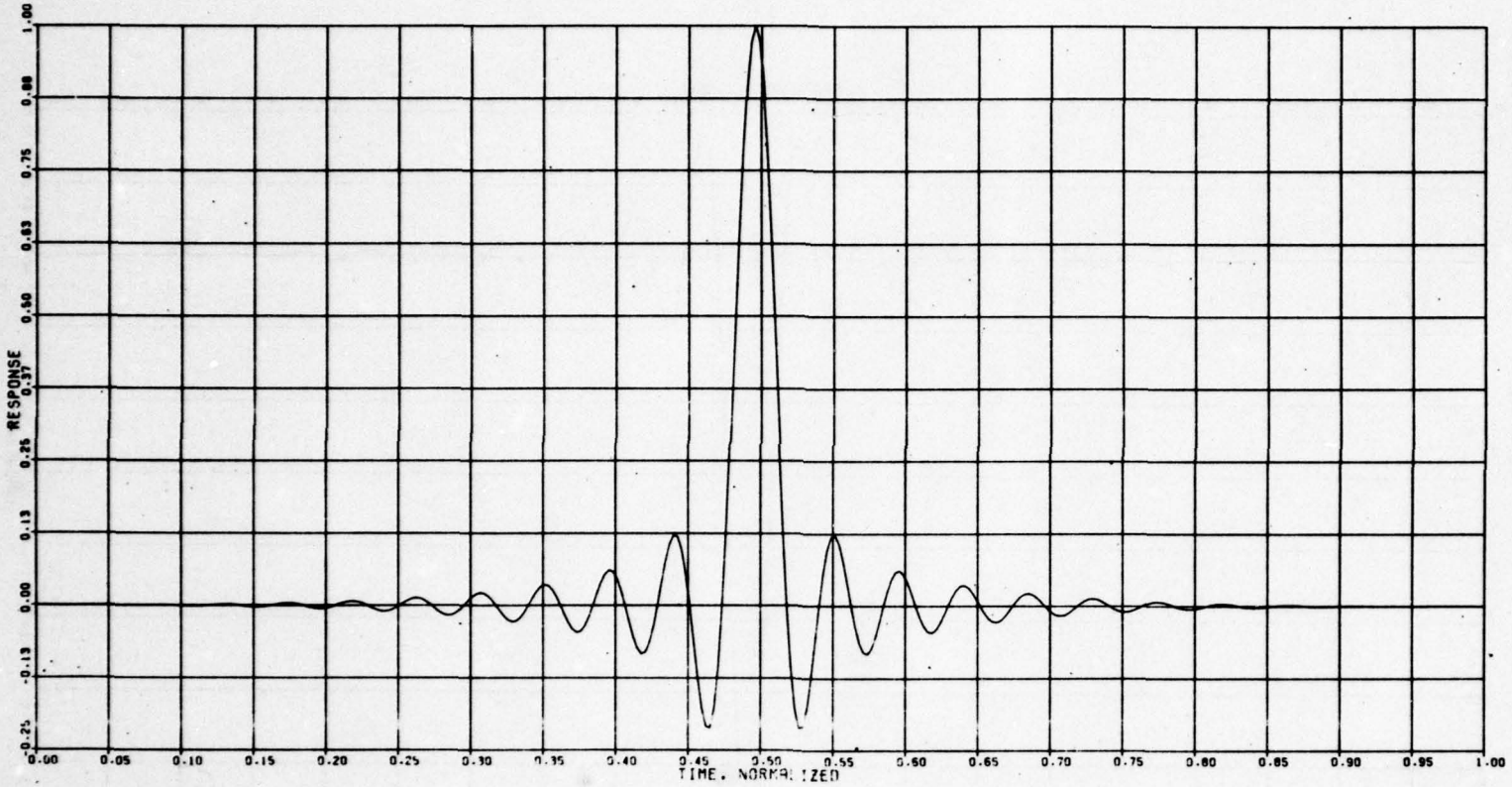


Figure 4-15. Filter Impulse Response, NT = 5, LEV = 3, N_f = 171. Frequency Function is Figure 4-3

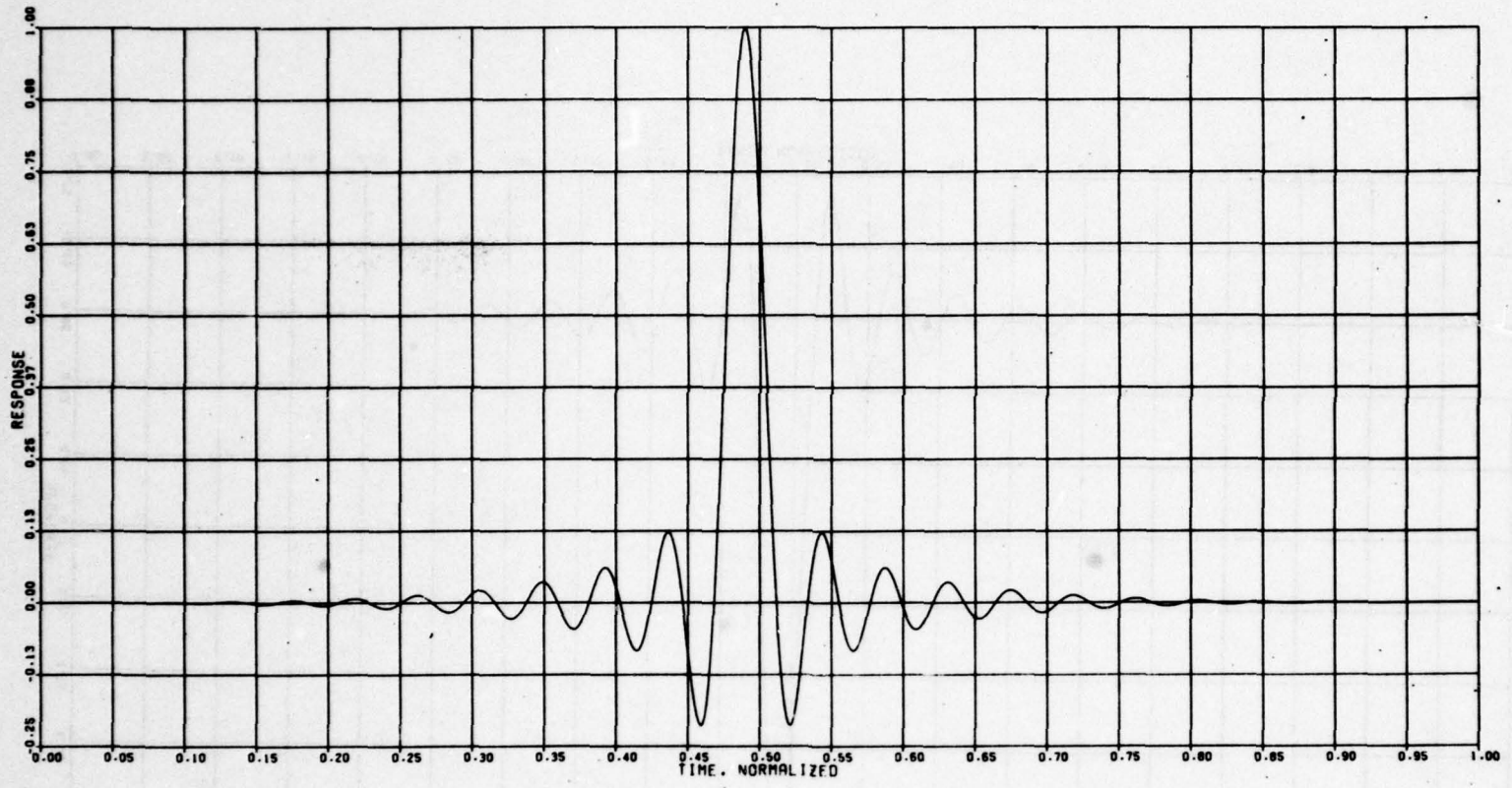


Figure 4-16. Filter Impulse Response, NT = 5, LEV = 4, N_f = 345. Frequency Function is Figure 4-4

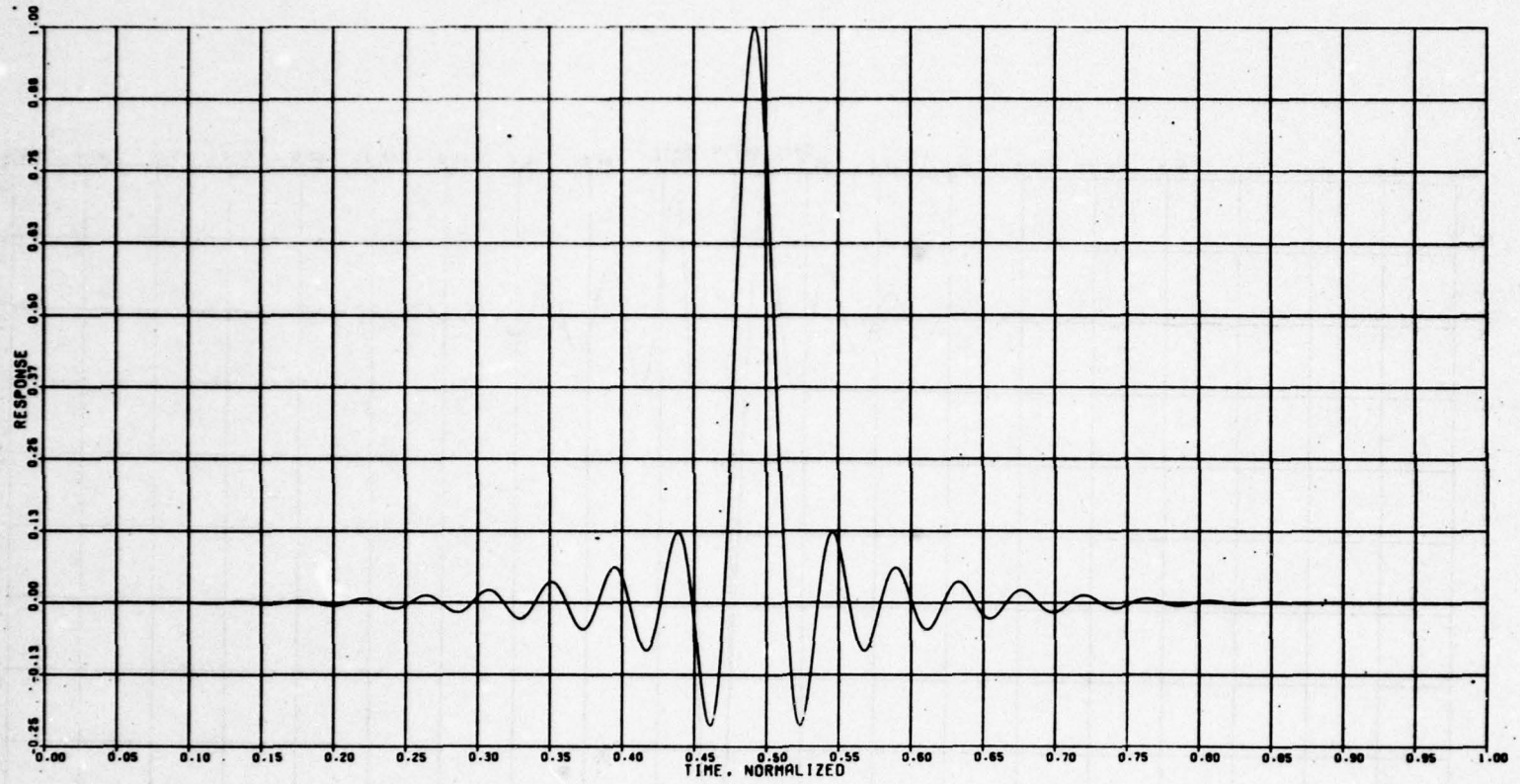


Figure 4-17. Filter Impulse Response, NT = 5, LEV = 5, N_f = 693. Frequency Function is Figure 4-5

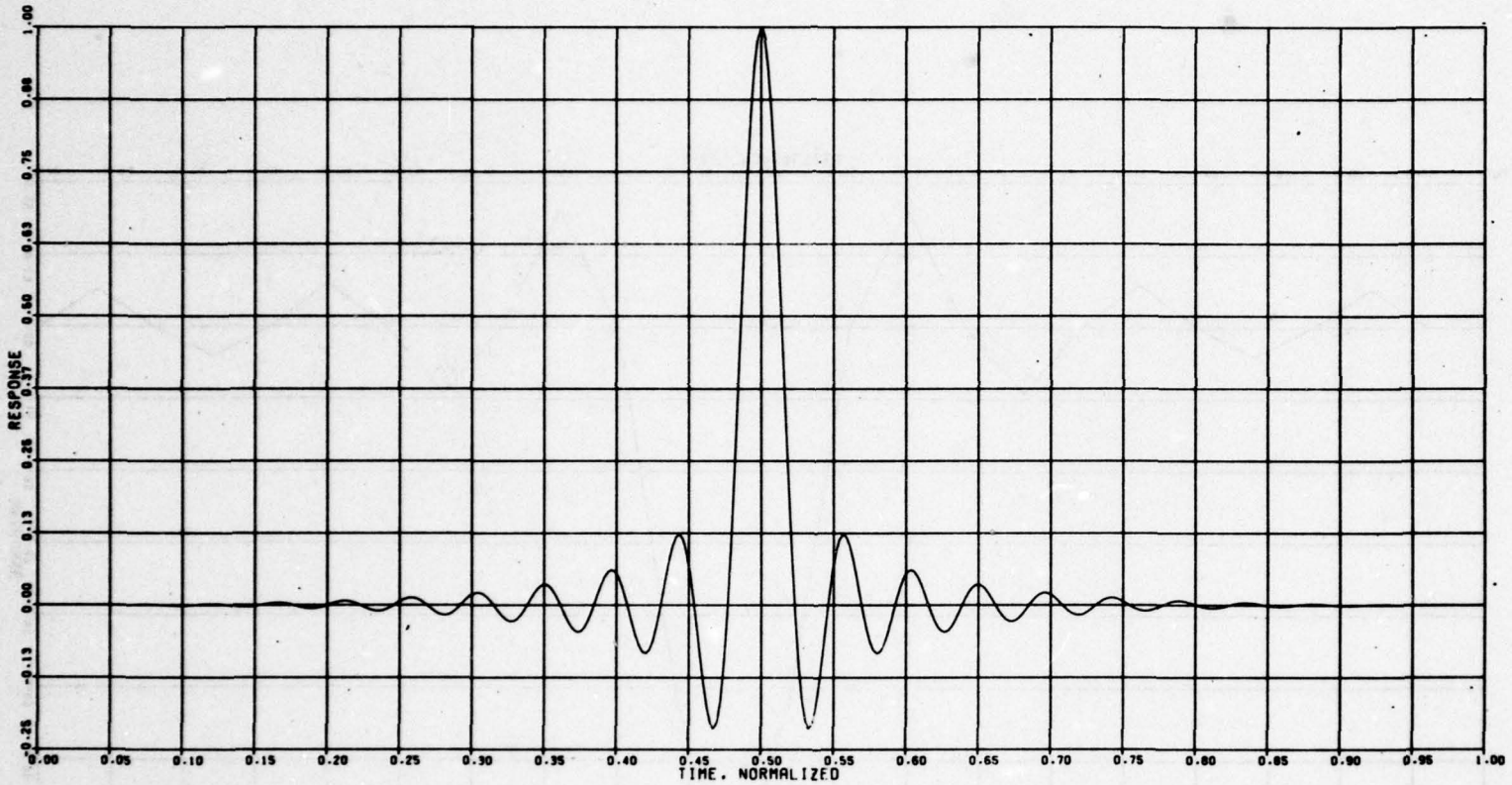


Figure 4-18. Filter Impulse Response, NT = 5, "J-Derived" From LEV = 7 Filter Using J = 13. N_f = 213. Frequency Function is Figure 4-11

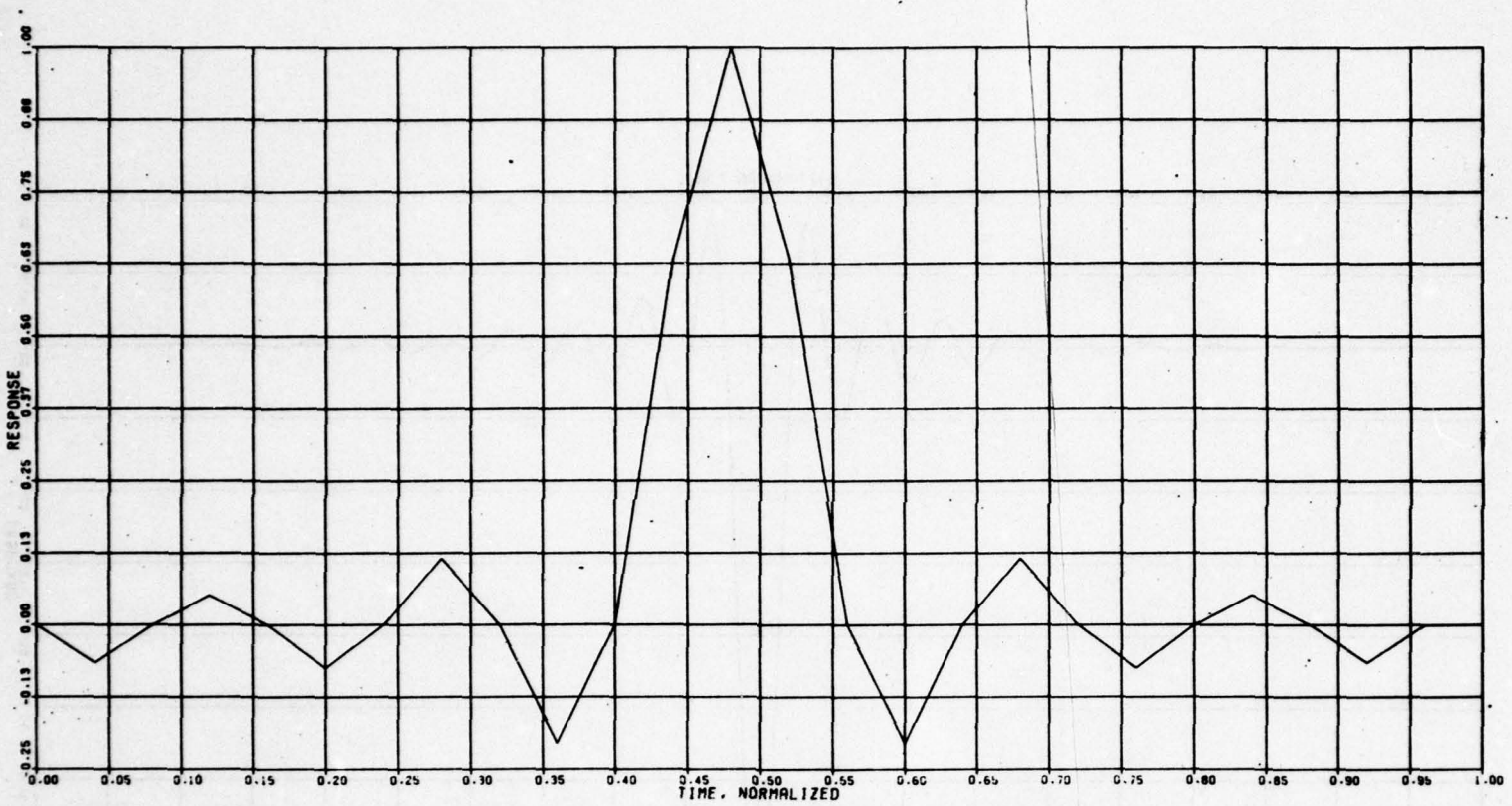


Figure 4-19. Filter Impulse Response, NT = 8, LEV = 1, $N_f = 11$

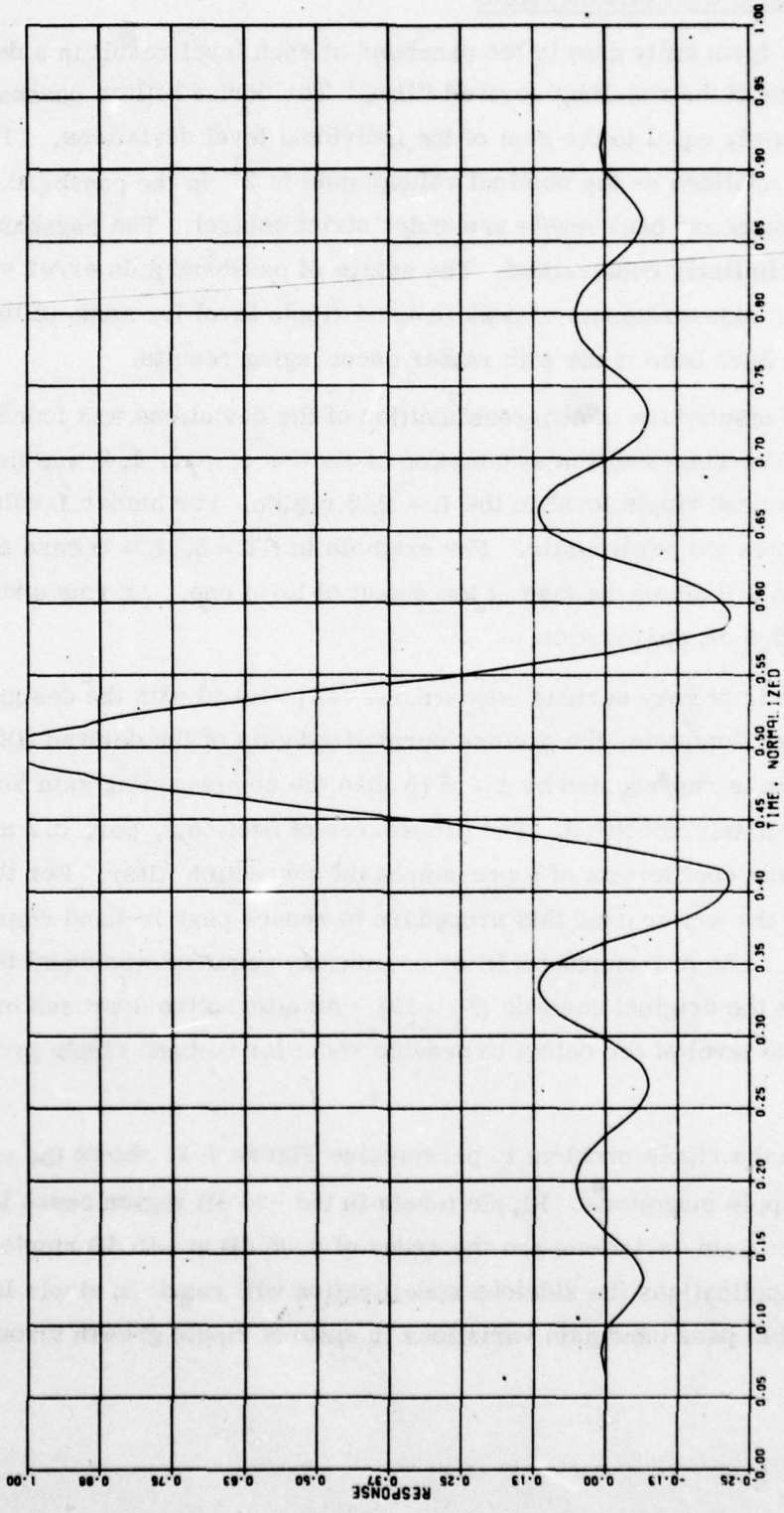


Figure 4-20. Filter Impulse Response, NT = 8, LEV = 7, N_f = 958

4.6 PASSBAND RIPPLE CONSIDERATIONS

Small deviations from unity gain in the passband at each level result in a deviation from unity in-band gain of the resultant derived filter. The derived filter passband gain deviation is approximately equal to the sum of the individual level deviations. (The derived filter gain must be normalized as the nominal voltage gain is 2^L in the passband.) In this design procedure the stopband peak levels are under strict control. The passband peak ripple levels are not similarly constrained. The nature of passband gain error cumulation is somewhat complex. Measurements of peak in-band ripple level for some of the derived filters discussed here have been made with rather encouraging results.

The worst-case assumption of coherent addition of the deviations was found to be far too pessimistic. The rms-addition assumption of Section 4, par. 4.4 was found to slightly understate the peak ripple level in the L = 2, 3 region. For higher L values the rms assumption becomes too pessimistic. For example in NT = 5, L = 7 case the peak in-band ripple is about 5.8 dB worse than at the output of level one. An rms addition assumption would predict an 8.5 dB degradation.

If in-band ripple is of very serious concern one can proceed with the designs as outlined previously. When complete, the precise normalized gain of the derived filter may be calculated. If this gain is represented by $1 + \delta(f)$ then the compensating gain function $1 - \delta(f)$ may be immediately specified. The procedures of Section 2, par. 2.2 may then be used to determine the coefficients of a pre-emphasis correction filter. For the specific case of NT = 5, L = 7 the writer used this procedure to reduce peak in-band ripple to level one (stopband) values. The pre-emphasis filter complexity required was equal to that of the first level filter in the original cascade ($N_f = 19$). An alternative approach might be to overspecify stopband level at the outset to provide room for in-band ripple growth through the cascade.

In order to keep the ripple problem in perspective Figure 4-21 shows the variation of pass band gain with ripple magnitude. Ripple levels in the -20 dB region cause less than a 1-dB gain variation. Gain variations are the order of 0.05 dB at -40 dB ripple levels. In many engineering applications the sidelobe specification will result in ripple levels which produce negligible pass band gain variations in spite of ripple growth through the cascade.

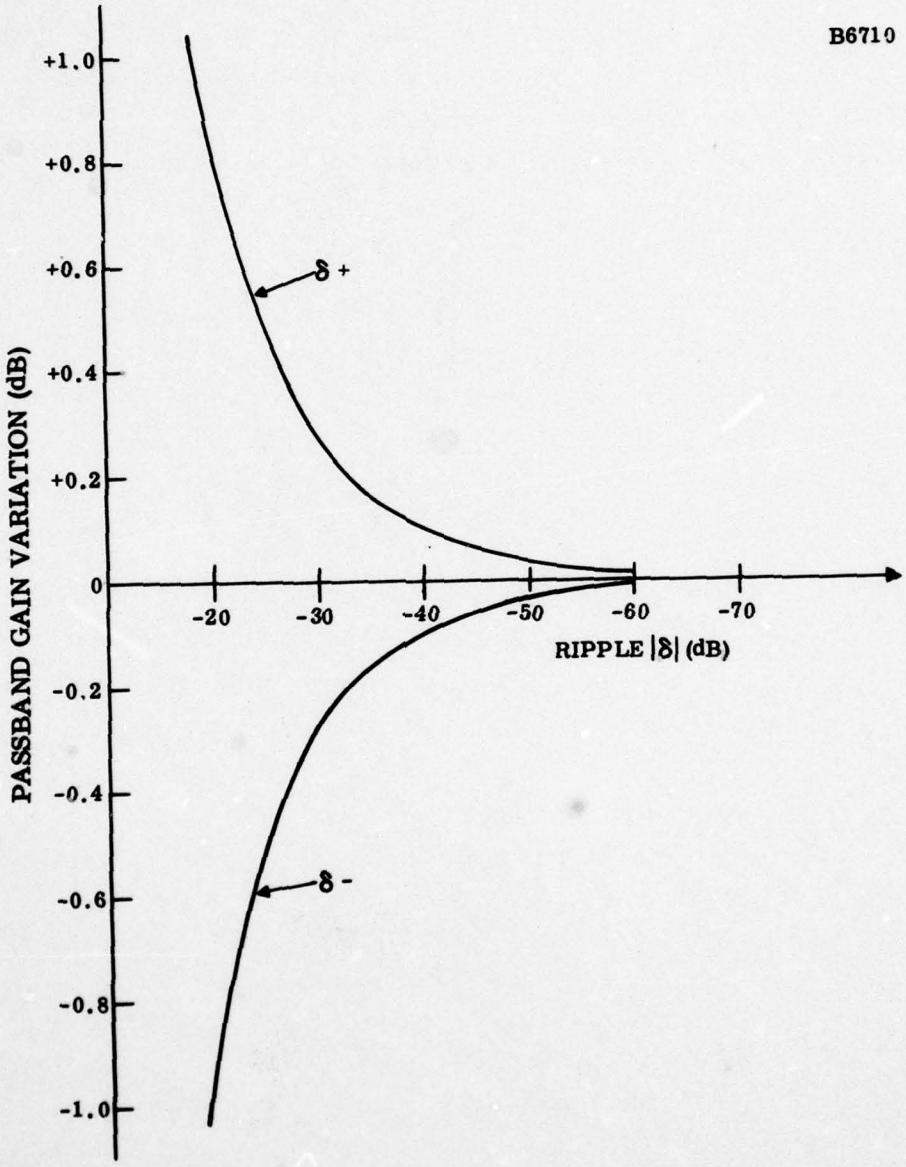


Figure 4-21. Passband Gain Variation as a Function of Ripple Value

SECTION V
APPLICATIONS

5.1 MODE CONVERSION, COMPLEX-TO-REAL

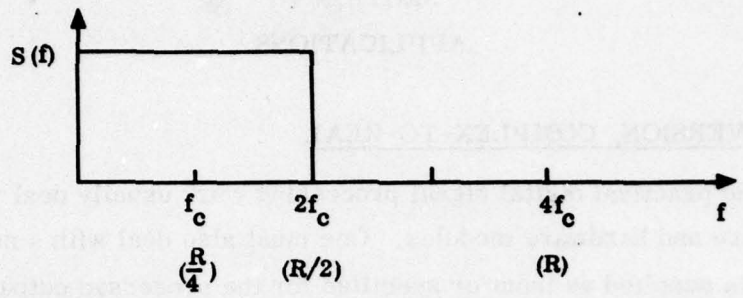
Those who do practical digital signal processing work usually deal with a variety of processing software and hardware modules. One must also deal with a number of different formats in the data supplied as input or specified for the processed output data. One of the most significant format characteristics concerns the nature of the sampling process itself. The individual sampling operation may involve one real number (a "real" sample) or an ordered pair of real numbers (a "complex" sample).

Consider the low-pass signal spectrum illustrated in Figure 5-1(a). The signal shown has an upper frequency limit of $2f_c$ so that the sampling rate R must be $4\ f_c$ if the usual single-number-per-sample ("real") low-pass sampling is done. It often happens, however, that the signal band of interest is centered at a frequency which is high compared to the bandwidth of the signal. In this case sampling at a rate equal to twice the highest frequency is very inefficient and one of two alternate procedures is normally followed.

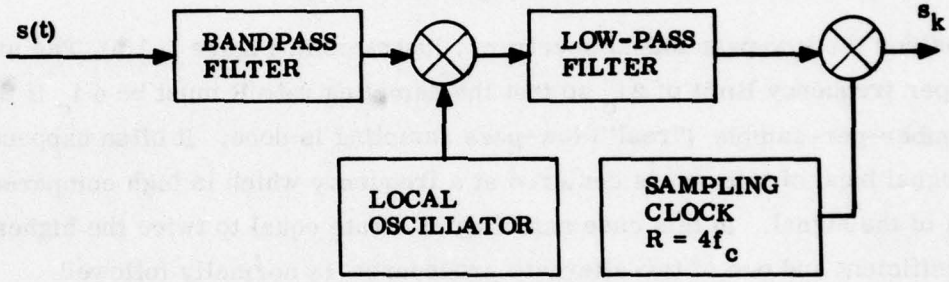
For purposes of illustration assume that the bandpass signal has a bandwidth of $2f_c$. The signal could be bandpass filtered then frequency-shifted down to a new center frequency, f_c, as shown in Figure 5-1(b). At this point the usual low-pass (real) sampling could be made at rate $4\ f_c$. This procedure requires a front-end filter tailored to the bandpass signal spectrum, a mixer, local oscillator, and post-mixer low-pass filter as shown in Figure 5-1(b).

An alternate approach, shown in Figure 5-1(c), uses product detection by cosine and sine signals from a local oscillator at the band center frequency, followed by low-pass filters to produce the familiar i (t) and q (t) signals. (In the general case these are not Hilbert transform related.) These in-phase and quadrature signal components have a high frequency of f_c. These signals are then each sampled at a rate $2f_c$ to produce, at each sample time, an ordered pair of real numbers (i_k, q_k). This pair is often referred to as a "complex" sample.

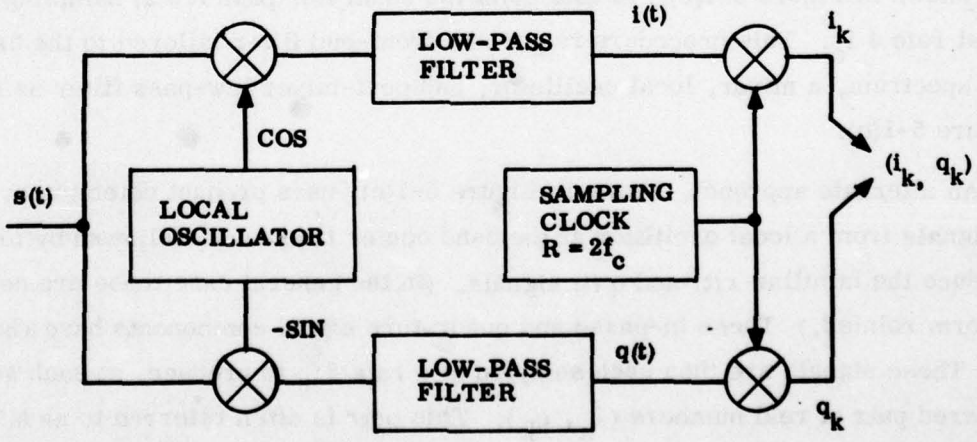
These two sampling techniques are theoretically equivalent. Each has its own practical problems and advantages and both are commonly encountered in practice. Data processing modules are not always designed to accept or produce either format. Hence one aften encounters the practical need for mode conversion from real to complex or from complex to real. The complex-to-real conversion will be discussed first with the aid of Figure 5-1.



a. Lowpass Signal Spectrum



b. Frequency Downshift Followed by Baseband Sampling



c. Complex (Bandpass Sampling)

Figure 5-1. Sampling Techniques

It may be seen from Figure 5-1(c) that three quantities define the bandpass signal: the center frequency, i(t), and q(t). If frequency translation of the signal is properly done, i(t) and q(t) remain unchanged and only the center frequency changes. Because of this invariance it may be assumed that the bandpass signal has been shifted to the center frequency f_c of Figure 5-1(a). The resulting time function may then be expressed both in terms of real samples and complex samples involving i(t) and q(t) (which are invariant with frequency translation). Relationships between real and complex sampling may therefore be established and used in mode-conversion operations.

The signal function s(t) may be expressed as

$$s(t) = i(t) \cos 2\pi f_c t - q(t) \sin 2\pi f_c t \tag{5-1}$$

A sampling rate of $R = 4 f_c$, will yield time values of $t_k = k/4 f_c$ so that Equation (5-1) becomes

$$s_k = i_k \cos \frac{\pi}{2} k - q_k \sin \frac{\pi}{2} k . \tag{5-2}$$

which means that

$$\left.\begin{aligned} s_k &= (-1)^k i_k, \qquad k = 0, 2, 4, 6, \text{---} \\ s_k &= (-1)^{\frac{(k-1)}{2}} q_k, \quad k = 1, 3, 5, 7, \text{---} \end{aligned}\right\} \tag{5-3}$$

It may be seen that the real samples s_k alternately reflect the i(t) and q(t) component values. Complex samples are taken at half the rate, but i(t) and q(t) are sampled simultaneously. If p_k represents a complex sample as per Figure 5-1(c) then it follows that

$$\left.\begin{aligned} \{s_k\} &= i_o, -q_1, -i_2, q_3, i_4, -q_5, -i_6, q_7, \text{---} \\ \{p_k\} &= \begin{matrix} i_o, & -, & i_2, & -, & i_4, & - & i_6, & -, & \text{---} \\ q_o, & -, & q_2, & -, & q_4, & - & q_6, & -, & \text{---} \end{matrix} \end{aligned}\right\} \tag{5-4}$$

If complex samples $\{p_k\}$ are given and real s_k samples are desired, Equation (5-4) shows that the even samples of s_k are directly available from the i_k terms of p_k. The odd samples of s_k are samples of q(t) but at times intermediate to the sampling times of the p_k number pair.

A single level of interpolation of the q terms of the complex samples is required. Only the interpolated values are to be kept, the original input q terms are discarded. The delay inherent in the interpolation operation on the q sequence must be applied to the i_k terms, and the sign changes in the s_k series of Equation (5-4) must also be accounted for. A subroutine for complex-to-real conversion based on the above principles appears in the Appendix as part of C432 and is also described in Section VI.

5.2 MODE CONVERSION, REAL TO COMPLEX

In Equation (5-4) it is now assumed that the sequence $\{ s_k \}$ is given and that the sequence $\{p_k\}$ is desired. For the sake of variety only, synthesis of the complex sequence at odd-numbered times will be considered. The q portions of the complex pair are available directly from the odd s_k terms as shown in Equation (5-4). The odd-numbered i terms must be generated by single-level interpolation from the even-numbered i samples. Only the interpolation data is kept, the input i data is discarded. Delay compensation of the q data to match the interpolator delay must be provided, along with proper sign-change accounting. A subroutine for real-to-complex conversion based on the above principles appears in the Appendix and is described in Section VI. For a different approach to the real-to-complex conversion see reference 11.

Conversions of the types discussed above involve an inherent phase ambiguity which is an integral multiple of $\pi/4$.

5.3 COMPARISON OF INTERPOLATION METHODS

Schafer and Rabiner (reference 3) discuss in detail interpolation using digital low-pass or stopband filters. In Section C of the above reference, comparisons to classical interpolation methods such as those of LaGrange are examined. These writers conclude that the frequency response characteristics of the classical interpolators leave much to be desired. They further conclude that for most digital signal processing work low-pass or bandstop filters are to be preferred over classical interpolation algorithms.

Attention can then turn to a comparison of the iterative technique of Figure 3-1 and the use of low-pass filters for interpolation. It has been shown in Section IV that the filters derived from the interpolation scheme of Figure 3-1 appears to be reasonably competitive with conventional filter designs in terms of processor demand. It is then possible to estimate the relative performance of the iterative interpolation method and the filtering method by use of Equations (4-1) and (4-2).

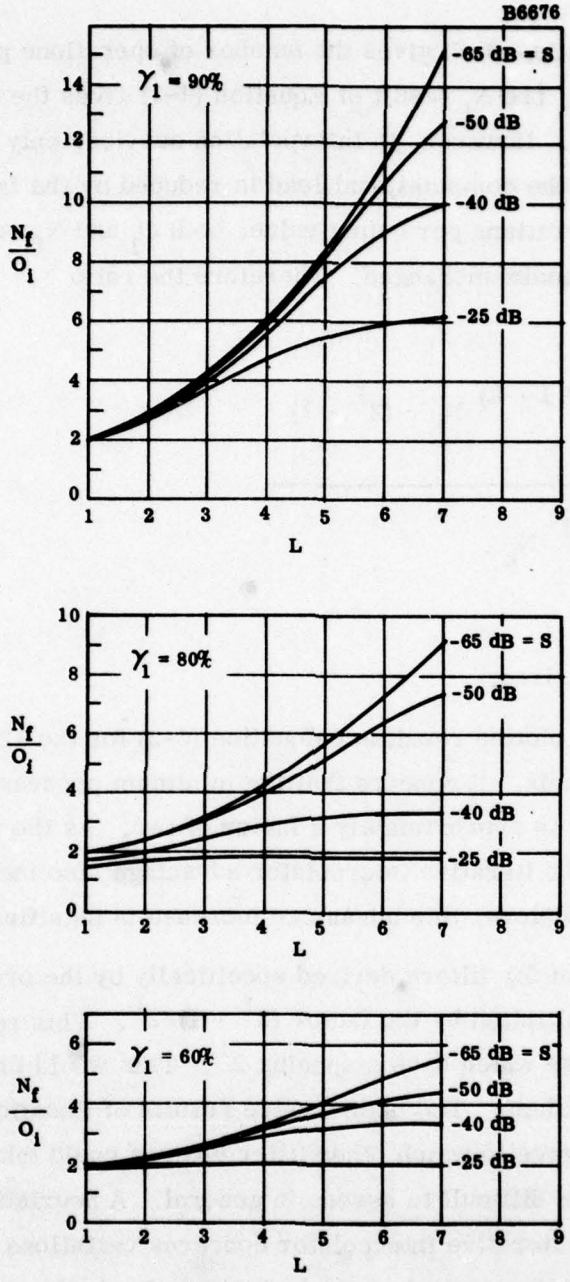
The O_i result of Equation (4-2) gives the number of operations per input value which produces 2^L output values. The N_f result of Equation (4-1) gives the number of filter operations per output value. However, in interpolation service, only one in every 2^L input samples is nonzero so that the computational load is reduced by the factor 2^L. If the comparison is based on operations per output value, both O_i and N_f would be divided by 2^L and their ratio would remain unchanged. Therefore the ratio

$$\frac{N_f}{O_i} = \frac{\sum_{k=1}^{L} 2^{(L+1-k)} N_k - (2^L - 1)}{\sum_{k=1}^{L} 2^{k-1} N_k} \qquad (5\text{-}5)$$

provides the comparison desired.

Figure 5-2 shows the plotted results of Equation (5-5) for the 12 interpolator designs of the program in the Appendix. It appears that the minimum processor demand advantage of the iterative interpolator is approximately a factor of two. As the sampling rate multiplication factor increases the iterative interpolator advantage also increases. For the higher-performance interpolators, this advantage increase is significant.

(It could be argued that for filters derived specifically by the procedures of this paper Equation (5-5) should be multiplied by the factor $(2^L - 1)/2^L$. This results from the exact nulls in the impulse response which have a spacing 2^L. This would bring the two procedures into near parity for low L values. The high L value results of Equation (5-5) however would remain unchanged. The degree to which other filter designs could take practical advantage of impulse-response nulls is difficult to assess in general. A heuristic explanation of the advantage of the Figure 3-1 iterative interpolator concerns variations of processor complexity and data rate through the cascade. Early in the cascade the processors are the most extensive, but the data rate is lowest. Later in the cascade when the data rate is high, the processors are short.) The advantages of the cascade over the lowpass filter for interpolation service are also discussed in references 8 and 10.



L = LOG_2 (SAMPLING RATE MULTIPLICATION FACTOR)
S = STOP BAND LEVEL (dB)
N_f = REQUIRED FILTER OPERATIONS
O_i = REQUIRED INTERPOLATOR OPERATIONS

Figure 5-2. Processor Demand Comparisons, Lowpass Filter vs. Iterative Interpolator

Table 4-1 gives some parameters of the interpolators and derived filters for levels from 1-5. The delay and N_f tabulations may be used as an aid in locating coefficients in memory when the Appendix program is used to provide equivalent filter impulse response.

5.4 ARBITRARY SAMPLING RATE MULTIPLICATION FACTORS

It often occurs, of course, that the sampling rate multiplication factor K is other than a power of two. The Appendix program, as written, will only give powers of two for K. There are at least two solutions to this problem.

The first approach is that used in the "J-derived" filter of Section IV. The program is run with some selected level L to yield a multiplication factor of 2^L. The resulting output data is then resampled with period J. The problem becomes one of selecting L, J so that

$$K \tilde{=} 2^L / J \tag{5-6}$$

A simple solution involves selecting a trial L and solving for J according to

$$\left.\begin{array}{l} J = N = \left[\dfrac{2^L}{K} \right] \\ \text{unless} \\ 2^L \dfrac{(2N+1)}{N(N+1)} - 2K > 0 \\ \text{then} \\ J = N + 1 \end{array}\right\} \tag{5-7}$$

where [ ] represents the integer part of the enclosed quantity. This trial L, J will yield an approximate multiplication factor, $\hat{K}$, of

$$\hat{K} = \frac{2^L}{J} \tag{5-8}$$

An error measure Δ_K may be defined

$$\Delta_K \triangleq \frac{\hat{K} - K}{K} = \frac{2^L}{JK} - 1 \tag{5-9}$$

One starts with the smallest L value which is at least as large as $\log_2(K)$. Equations (5-7), (5-8) and (5-9) are used and the resulting Δ_K is noted. If the error is too large, increment L by one and repeat until the error reaches an acceptable level. An HP-67 program for implementing this procedure is given in the Appendix, and labeled "PROGRAM 1-11-79".

For example let K = 13. The above procedure yields a solution L = 9, J = 39 for 1% error ($2^9/39 = 13.13$), or L = 10, J = 79 for 0.3% error ($2^{10}/79 = 12.96$). This approach will always yield the minimum L required to meet any output sampling rate multiplication factor error specification.

This technique will work very nicely also when the desired output sampling rate is not an integer multiple of the input sampling rate. For example let a sampling rate multiplication factor K = π be required within 0.02%. The above procedure quickly yields L = 9, J = 163 ($2^9/163 = 3.1411$, $\Delta_K = 0.0155\%$).

The second solution to this problem involves a modification of the computer program. One must traverse enough processors of the Figure 2-3 type until the sampling rate is sufficiently high so as to permit linear interpolation. Once this stage is reached the sampling rate multiplication factor at the linear interpolation level can be any integer value. Due to a foolish consistency (hopefully not that of which Emerson spoke) the writer continued the power of two logic through the last "stage" when coding the Appendix program.

Hindsight now makes it clear that this stage should have been coded for any desired multiplication factor, say J. The overall sampling rate increase could then be $2^L \times J$ where L is the number of Figure 2-3 doubler stages. The output data could then be censored by a factor of 2^L to give precisely a 1:J overall factor. This feature would also lend more flexibility to the operations discussed relative to Equations (5-6) through (5-9).

5.5 SAMPLING RATE DIVISION

The advantages of the cascade interpolation technique of Figure 3-1 over the classical low-pass filter processing of zero-filled input data have been discussed in detail. If one studies Figure 3-1 and visualizes 2^L output samples for each input sample, simple faith in the consistency of nature raises, on first impulse, the possibility of putting 2^L samples in at the bottom and getting one sample out at the top. Is there a cascade equivalent of the conventional low-pass filter/data censor combination? The answer is yes and, furthermore, the significant advantages of the cascade over the low-pass filter method noted for interpolation are again to be enjoyed in the sampling rate division operation. a FORTRAN program for data rate division also appears in the Appendix under deck name C434.

(The advantages of multistage data rate reduction (decimation) are discussed in references 6, 8 and 9. In the case where lowpass filtering is to be done while maintaining the sampling rate, reference 9 demonstrates significant gains for a joint use of rate reduction followed by rate multiplication.)

The sampling rate division cascade uses the same level processor-equivalent filters as the interpolation cascade but with some significant differences. The processors are used in reverse order. The exit is always from the level one processor, and the entrance depends on the number of levels or L value of the rate-reduction operation. For example, if a 16:1 rate reduction is specified, L = 4 and the cascade would be entered at the level four design where a 2:1 rate reduction would be achieved after filtering. This output would then go to a level 3 design followed by a level 2 design, and finally to the level 1 design for pre-output processing. If L exceeds 5, a linear interpolation processor leads off the chain, but this is a special case which will be treated separately later.

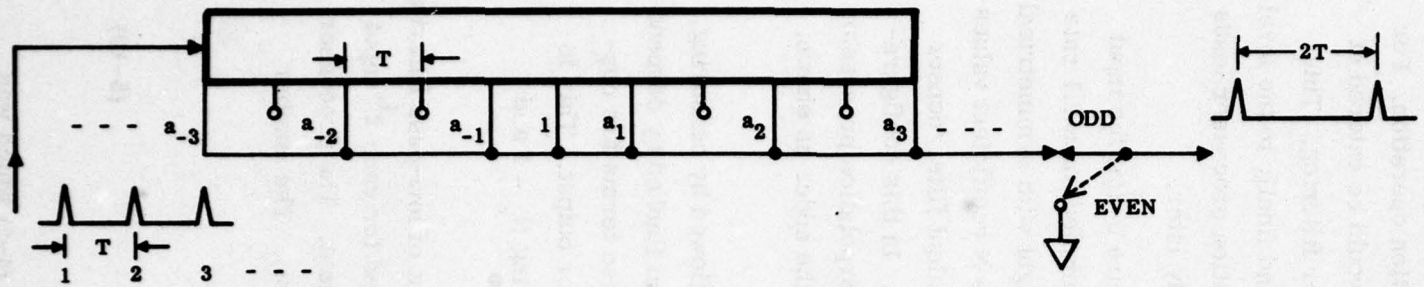
Figure 5-3 illustrates the process at each normal level. In Figure 5-3(a) the input data at rate R is passed through a low-pass filter whose output is resampled for a 2:1 rate reduction. As discussed earlier, these filters have a unit central weight with symmetrical filter coefficients and interleaved zero weights as shown. If there are N coefficient values for this filter, (4N-1) storage locations are required. An exact equivalent filter/censor combination requiring only (3N+1) locations is shown in Figure 5-3(b). In this configuration the input values are switched between what turns out to be the interpolation processor and a delay element. The censored output is available directly from the adder as shown. It is this latter processor that is coded in C434.

If L exceeds 5 the lead-off processor is a linear interpolator followed by censoring. The impulse response of the linear interpolator has a unit central term flanked by descending sized terms equal to $M/2^{L-5}$ where M = 1, 2, --- $(2^{L-5} - 1)$. These terms are contiguous. Following the interpolator, every 2^{L-5}th value is selected for output. This is precisely equivalent to a chain of L-5 normal cascade processors having $N_k = 1$ and $a_1 = 1/2$.

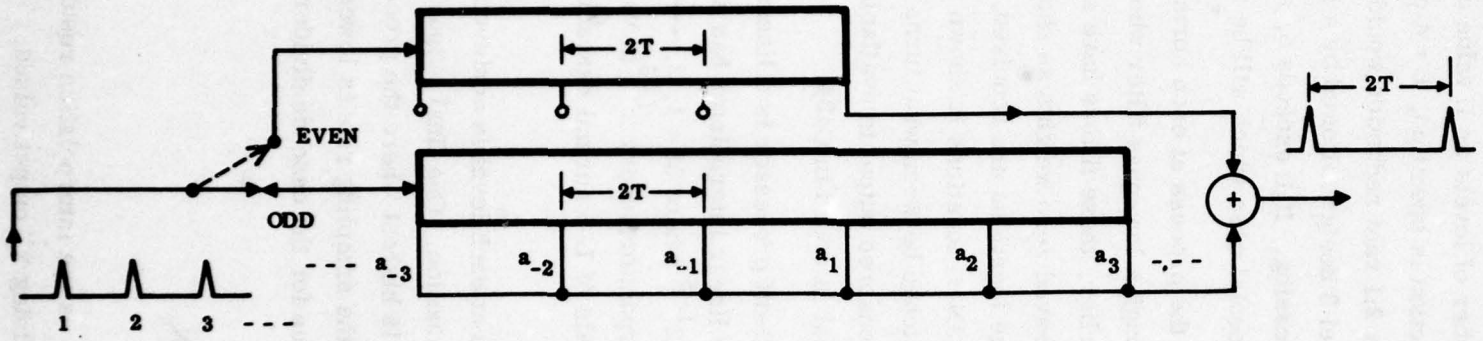
Sampling rate division at each level is achieved by the equivalent of low-pass filtering followed by 2:1 data rate reduction. One final output value is generated for each 2^L input values. The sampling rate is highest where the processors are shortest. The processor lengths are greatest where the sampling rate is lowest (near the output). The number of operations per output value for the cascade divider is

$$O_d = \sum_{k=1}^{L} 2^{k-1} N_k \qquad (5\text{-}10)$$

which is precisely the same as the interpolation result O_i of Equation (4-2) which was based on one input pulse yielding 2^L output values.



(a) Cascade Processor Low-Pass Filter Equivalent With Output Censor



(b) Cascade Processor Low-Pass Filter Equivalent With Input Switching

Figure 5-3. Data Rate Division Cascade Divide-By-Two Level Details

The delay through the cascade divider measured in input pulses is (see also Equation (4-1)).

$$D_d = \sum_{k=1}^{L} N_k 2^{L-k+1} - (2^L - 1) \qquad (5\text{-}11)$$

whether dealing with interpolation or rate reduction, the equivalent overall low-pass filter is the same for a given (NT, LEV) pair. It is not surprising then to find that the ratio of operations required for filtering divided by the number of cascade operations per output data value is, for rate reduction,

$$\frac{N_f}{O_d} = \frac{\sum_{k=1}^{L} N_k 2^{L-k+1} - (2^L - 1)}{\sum_{k=1}^{L} N_k 2^{k} - 1} \qquad (5\text{-}12)$$

which is exactly the result of Equation (5-5) for interpolation. It follows therefore that the considerable-computational efficienty advantages of the cascade over the filter shown in Figure 5-2 for interpolation apply here also. Note also that the only limit to rate reduction factor in the present program is the 2^L memory size required of the input array.

The impulse response determination of the equivalent low-pass filter obtained by driving the interpolation program C432 with a unit value followed by a string of zeros will not work with the rate reduction program C434. What one obtains in the latter case is a 2^L-censored version of the relevant impulse response; not a very useful result. This is perhaps a manifestation of the reversibility of the interpolation process as compared to the irreversibility of the rate-reduction process. The latter discards information, the former does not.

Cascade-derived impulse responses were used successfully and correctly in the interpolation case, but care must always be exercised because of the sampling rate changes through the cascade. Even in C432 the impulse response of a two-level cascade is not the normal convolution of the individual level impulse responses. If one is intersted in equivalent filter impulse responses program C432 must be used; C434 cannot be employed for such a purpose.

The low-pass filter equivalents of the individual level processors have very useful spectral properties for the two-to-one rate reduction achieved in each stage. In our frame of reference, frequency extends from zero to unity (referenced to R/2 or half the sampling rate). The symmetric coincidence of passband and stopband about f = 1/2 for these filters is fortuitous. The spectral components in the stopband region will be folded into the passband. Hence the region of high attenuation is precisely where it will do the most good.

The spectral components near frequency f = 1/2 are attenuated by about 6 dB (transition region center) and are moved up to f = 1 for the next stage. In the next stage stopband attenuation is applied to this reduced level aliasing band before being folded into the passband. The cumulative attenuation effects of the transition regions as the cascade is traversed keep the passband noise level growth well under control for high L values.

Design of a rate division processor is quite simple. The equivalent filter transition region width and stopband attenuation are defined by the NT choice. The rate-reduction factor 2^L (via specification of LEV in C434) may be thought of as defining a filter at input level having passband edge $f_p = \gamma_1/2^L$ where γ_1 refers to the interpolator level one processor (which is now at the end of the rate-division chain). At the final cascade output, censoring will have moved f_p to the value γ_1.

As a test of the C434 concept, programs LINT and LDIV were run back-to-back for NT = 5 and LEV = 7. That is, a 1:128 data rate expansion is first done and is followed by a 128:1 data rate reduction. The net transfer function result should be constant within the passband constraints of the NT = 5 design. A unity-value sample followed by a string of zeros was entered into LINT to produce the impulse response whose frequency function is that of Figure 4-7. This impulse response data was then entered into LDIV. Note that one data point into LINT produces one data point out of LDIV with 128 points being involved each time at the C432/C434 interface.

This operation produced 87 nonzero final output values which are plotted in Figure 5-4. The Fourier transform of this overall impulse response (8192-point FFT) is shown in Figure 5-5. The response is flat out to f = 0.9 where gain begins to drop to a -11 dB value (?) at f = 1. Sampling rate division expands the frequency scale between input and output. Note that the input to LDIV had a band edge value $f_p = 0.00703125$ (see Figure 4-7 caption); when this f_p value is multiplied by 128 an $f_p = 0.9$ results.

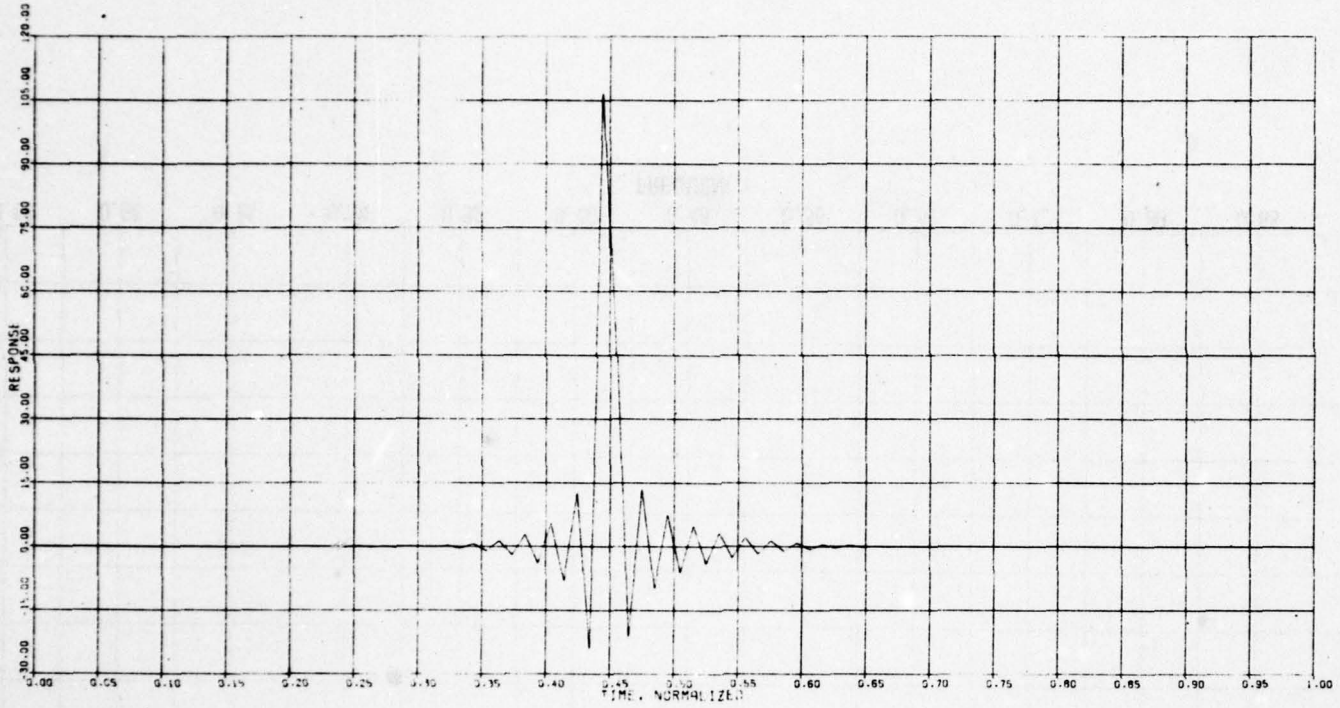


Figure 5-4. Impulse Response of LINT, LDIV Chain for NT = 5, LEV = 7. (There are 87 nonzero points.)

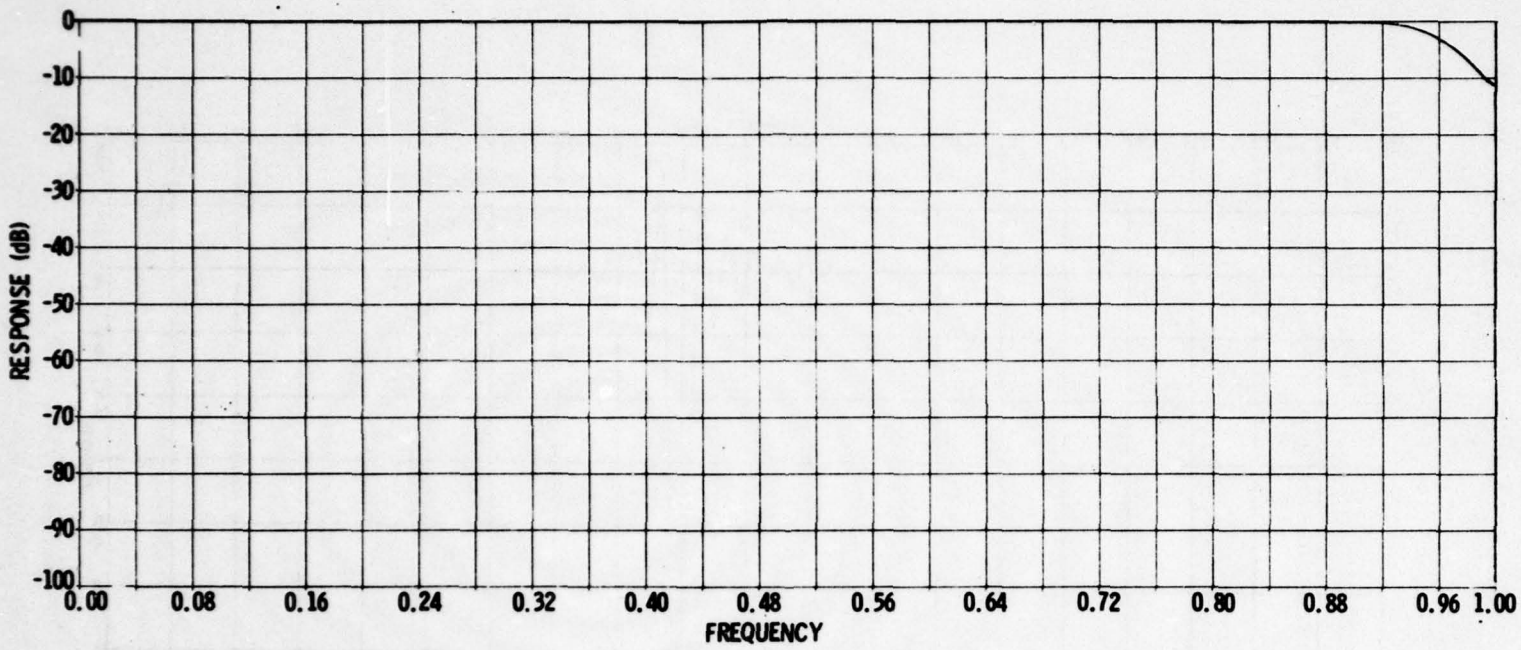


Figure 5-5. Frequency Response of LINT, LDIV Chain for NT = 5, LEV = 7. (This is a Fourier transform of the Figure 5-4 time data.)

An expanded scale plot of the Figure 5-5 data is shown in Figure 5-6. When properly adjusted for normalization the "round-trip" passband your variation limits are +0.0125 dB at f = 0 and -0.0226 dB at f = 0.9.

If one were to obtain the frequency response curve of the divider chain plotted in terms of input frequency, the results would be precisely those obtained by taking the Fourier transform of the impulse response of the equivalent interpolator chain. The true output frequency would, of course, be 2^L times the input frequency. Let f_i be the input frequency and f_d be the divider output frequency (both normalized to R/2). Define

$$f_t = \left[2^L \times f_i \right]_{\text{MOD } 2} \tag{5-13}$$

then

$$f_d = f_t \text{ for } f_t \leq 1 \tag{5-14}$$

or

$$f_d = 2 - f_t \text{ for } f_t > 1 \tag{5-14a}$$

The frequency division factors available directly from C434 are powers of two. This restriction may be lifted by use of the interpolation program as a preprocessor prior to use of the rate division program. Let the division ratio desired be represented by V_d which is greater than one but is not a power of two. Let L_i, J_i be the interpolation program parameters as used in Equation (5-6) and related discussion in Section V4. par. 5.4. Let L_d be defined as the level specification for the C434 rate division program. The division ratio V_d is first specified. The L_d, L_i and J_i values are sought which will define the minimum-complexity interpolator/divider chain to achieve a division ratio V_d, plus or minus a tolerance specification.

First choose the smallest L_d such that

$$\frac{2^{L_d}}{V_d} > 1 \tag{5-15}$$

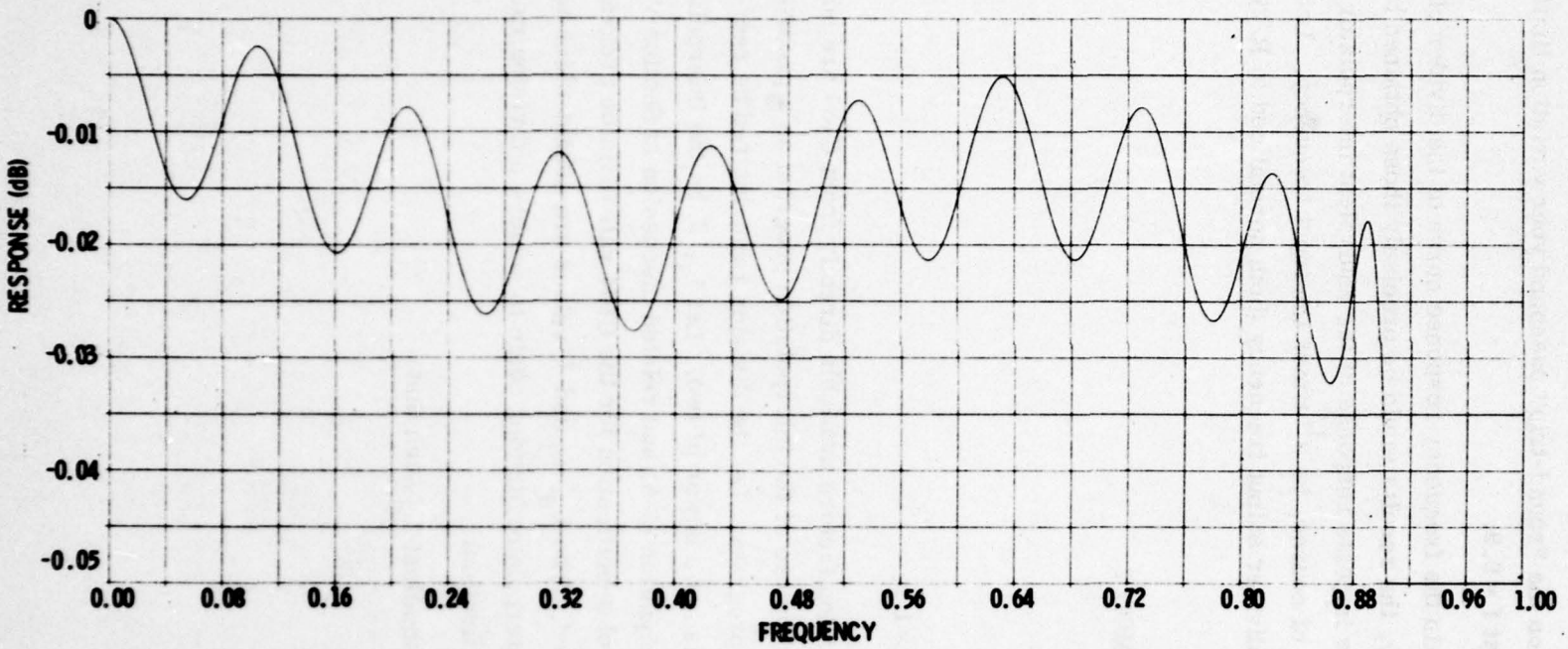


Figure 5-6. Frequency Response of LINT, LDIU Chain for NT = 5, LEV = 7 (Expanded-Scale Plot of Figure 5-5 Data)

If V_d is an integer, then

$$\left.\begin{aligned} L_i &= L_d \\ J_i &= V_d \end{aligned}\right\} \quad \left(\begin{matrix}\text{Integer} \\ V_d \text{ only}\end{matrix}\right) \tag{5-16}$$

and

which simply says to multiply the sampling rate first by the factor $2^{L_d}/V_d$ in C432, then divide this resulting sampling rate by 2^{L_d} in C434.

If V_d is not an integer, determine L_d first as in Equation (5-15) then define a rate multiplication factor K_i of

$$K_i = \frac{2^{L_d}}{V_d} \cong \frac{2^{L_i}}{J_i} \tag{5-17}$$

and use the procedures of Section V par. 5.4 to solve for L_i, J_i, the interpolation operation parameters.

The frequency-domain effect of interpolation is compression. A spectrum that extends from 0 to γ_1 (normalized to R/2) is compressed to 0 to γ_1/V_i: after a $1:V_i$ sampling rate increase. Information is not lost in this process. The frequency-domain effect of rate reduction is expansion after filtering. The 0 to γ_1 spectrum is first lowpass filtered with $f_p' = \gamma_1/V_d$ where V_d: 1 is the rate reduction ratio. After this filtering operation the spectrum is expanded by the factor V_d so that γ_1/V_d becomes γ_1. Signal information originally residing in γ_1/V_d to γ_1 is lost in the process.

An examination of the frequency-domain effects of the example of Equations (5-15) through (5-17) will show that the net effect of interpolation followed rate reduction is an equivalent rate reduction operation of ratio V_d : 1. If the rate reduction of ratio V_d precedes the interpolator of ratio V_i a much different result obtains. The original input signal is lowpass filtered with $f_p' = \gamma_1/V_d$. This cutoff frequency is then moved to γ_1 by the censoring operation. The resulting spectrum is then compressed by the interpolation ratio V_i. Frequency components in the original spectrum above γ_1/V_d are suppressed while components whose frequencies are below this cutoff value are relocated in spectral position by the factor V_d/V_i. If $V_d = V_i$ the net effect of this combined operation is lowpass filtering only. The input and output sampling rates will be equal and the passed spectral components retain their original frequency positions. As discussed in reference 9, the above arrangement is an efficient approach to the problem of lowpass filtering when sampling rate changes are not permitted.

SECTION VI

COMPUTER PROGRAMS

6.1 MULTILEVEL INTERPOLATION PROGRAM LINT

In the Appendix a computer program with deck name C432 is listed. Concurrent loading of the Block Data subprogram C435 is required. The program is written in Honeywell YFORTRAN time-share format. In this section the concern is with entry names LINT and the initializing entry XLINT, which must be called prior to the first LINT call.

CALL XLINT (NT, LEV, LAG)

NT - This is an input quantity and selects the interpolator type. Tables 4-1 and 4-2 give details. Types 1-8 assume significant spectral energy out to half the sampling rate, and use an extended γ at the second level in order to maintain low stopband responses. NT values of 9-12 should be used in the oversampled cases as the γ progression here is a straight one to one-half. Low signal spectrum levels are assumed in the γ (R/2) to (R/2) frequency range for NT = 9 through 12.

LEV - This input quantity is the number of levels of interpolation. The sampling rate will be multiplied by 2^{LEV}. There is no uppler limit to LEV. However the user must provide sufficient space in the output array, FØUT to contain 2^{LEV} values.

LAG - This output quantity represents the total delay through the interpolation processor expressed in terms of samples at the output sampling rate.

After XLINT is used, a call to the main entry LINT is made for each input data value

CALL LINT (VIN, FØUT)

VIN - This is a single input sample value.

FØUT - This is a vector in which the output sample values are returned by the program. 2^{LEV} values are returned for each input value.

6.2 MODE CONVERSION PROGRAM CØRE, RECØ, AND CØNV

CALL CØNV (NT, LAGP)

This is an initializing entry, and must be called before first use of either CØRE or RECØ. CØNV services both mode conversion routines.

NT - This is an input quantity and selects the interpolator type. Only the first three columns of Table 4-2 have any significance in this usage.

LAGP - This output quantity represents the mode-conversion processor delay in terms of output sample pairs.

The complex-to-real transformation is done by

CALL CØRE (CØMP, RE)

CØMP - A two-location input vector containing one complex input data pair.

RE - A two-location output vector containing a sequential pair of real samples.

The real-to-complex transformation is done by

CALL RECØ (RE, CØMP)

RE - A two-location input vector containing a sequential pair of real samples.

CØMP - A two-location output vector containing one complex output data pair.

6.3 HP-67 PROGRAM 1-5-79 FOR SECTION V, PAR. 5.3 EQUATIONS

This program evaluates Equation (5-5) but in the process it yields the delay and N_f value of Equation (4-1), the O_i value of Equation (4-2), the N_f/O_i ratio of Equation (5-5), and finally the N_c value of Equation (4-3).

1. Depress "D" to initialize. Program can handle level values (L) up to 15. All $\{N_k\}$ values are set to unity during this initialization. Wait for program halt.
2. Enter number of coefficients at each level starting with level one. Enter one value and depress "R/S".
3. Repeat step 2 till $\{N_k\}$ set is stored. Values beyond level entered are one.
4. Enter L value, depress "C".
5. Program halts with N_f, D_i value of Equation (4-1) in display. To continue depress "R/S".
6. Program halts with O_i of Equation (4-2) in display. When ready to continue, depress "R/S".
7. Program halts with N_f/O_i of Equation (5-5) in display. Use "R/S" to continue.
8. Program halts with N_c of Equation (4-3) in display. This completes the sequence.
9. At this point the user can return to step 4 to repeat with a new L value and the same $\{N_k\}$ set, or the user may return to step 1 to enter new $\{N_k\}$ set.

6.4 HP-67 PROGRAM 1-11-79 FOR SECTION V, PAR. 5.4 EQUATIONS

This program may be used to execute the procedure discussed in Section V, par. 5.4, specifically with reference to Equations (5-6), (5-7), (5-8), (5-9).

1. Enter desired sampling rate multiplication factor K. Depress "A".
2. Program halts showing next L value to be tried. Press "R/S" when ready for trial cycle.
   a) Program pauses (flashing decimal) and displays $|\Delta_K|$ % of Equation (5-9).
   b) Program pauses to display J value of Equation (5-7).
   c) Program pauses to display $\hat{K}$ value of Equation (5-8).
3. Program increments L and returns to step 2.

Note: During step 2, halt, user may enter any L value for trial. If user enters L value and "STØ2" L sequence is modified. For data recovery, incremented L is in register 2, last J value is in register 1.

6.5 MULTILEVEL SAMPLING RATE DIVISION PROGRAM LDIV

This program is listed under deck name C434 and requires concurrent loading of the Block Data subprogram C435. The program is written in Honeywell YFORTRAN time-share format. There are two entry points to this program. An initializing entry XLDIV must be called prior to the first usage of the working entry LDIV.

CALL XLDIV (NT, LEV, LAG)

NT - An input variable which selects the processor type. Type specifications 1 through 12 are currently valid.
LEV - An input variable which specifies the number of levels of rate division. The sampling rate will be divided by 2^{LEV}. There is no upper limit to LEV, however the user must dimension an input array FIN of size 2^{LEV} or greater.
LAG - This output variable represents the total delay through the rate division processor expressed in terms of input samples.

After XLDIV is called, a call to the main entry LDIV is made for each group of 2^{LEV} input values; one output value results for each LDIV call.

CALL LDIV (FIN, VOUT)

FIN - An input vector containing 2^{LEV} data values.
VOUT - A single output variable.

SECTION VII

CONCLUSIONS

The ease of design, computational economy, and high data-rate increase factors available from the iterative-interpolation technique make this approach attractive relative to the conventional filter method of sample-rate multiplication. The filter synthesis procedure which was derived from the interpolation algorithm also appears to have practical utility.

The computer program C432 of the Appendix should be useful in its present form for a variety of interpolation, filter design, and mode-conversion function. Some improvements in the program are planned, however. A $\gamma = 95\%$ cascade should be added. This gives a transition region which is 11% of the passband. Such designs are unavoidably expensive in processor length. If digital filters are ever to rival analog filters in performance, hardware technology must advance to the point where such designs can be accepted. The rate-doubling sections of the cascades should be increased in number to accommodate some lower stopband level specifications. As previously discussed, the linear interpolation stage should be recoded for an arbitrary 1:J sampling rate increase. The algorithms in Section V par. 5.4 for arbitrary sampling rate multiplication factors could be included in a future version of the program. The user would then be required only to specify the sampling rate multiplication factor desired and the program would automatically provide the optimum solution.

Notice should be taken of the real advantages of applying the iterative interpolation technique as opposed to the more conventional use of low-pass or band stop filters for this service. The ease with which good interpolators can be designed and their efficiency in service, especially at high multiplication ratios, should suggest many new areas of application for the interpolation process.

One such new area of promise concerns the use we have made here of the interpolator in filter design. Basically, a generic impulse response function, believed to be optimum in some sense, primes the process. Whether this generic function is obtained out of first level as in our example, is or supplied externally is incidental. The proper interpolation process is then applied to derive filters having passband frequency functions which are scaled replicas of the generic filter equivalent functions. It would be useful to know to what

degree the original optimality property is retained in the derived filters. If the degree of retention is high, these results could have significance beyond the digital filter design goals of Section IV.

The complementary use of the cascade technique for sampling rate division in program C434 makes available many of the advantages of the companion C432 interpolation program. A combination of these two programs offers the user a convenient, efficient set of procedures for a variety of signal processing tasks.

SECTION VIII

REFERENCES

1) Hamming, R. W., "Numerical Methods for Scientists and Engineers", Chapter 25, New York: McGraw-Hill, 1962.

2) McClelland, J.H., Parks, T.W., and Rabiner, L.R., "A Computer Program for Designing Optimum FIR Linear Phase Digital Filters" IEEE Trans on Audio and Electroacoustics, Vol. AU-21, pp. 505-526, December, 1973.

3) Schafer, R.W., and Rabiner, L.R., "A Digital Signal Processing Approach to Interpolation", Proc. IEEE, Vol. 61, pp. 692-702, June 1973.

4) Kaiser, J.F., "Design Methods for Sampled-Data Filters, Proc." First Allerton Com. on Circuit Systems Theory, pp. 221-236, 1963.

5) Bellanger, M.G. Daguet, J.L., and L.E. Pagnol, G.P., "Interpolation, Extrapolation, and Reduction of Computation Speed in Digital Filters", IEEE Trans. Vol. ASSP-22, No. 4, August 1974, pp. 231-235.

6) Shively, R.R., "On Multistage Finite Impulse Response (FIR) Filters with Decimation," IEEE Trans. Vol. ASSP-23, No. 4, August 1975, pp. 353-357.

7) Oetken, G., Parks, T.W. and Schussler, H.W., "New Results in the Design of Digital Interpolators", IEEE Trans. Vol. ASSP-23, No. 3, June 1975, pp. 301-309.

8) Crochiere, R.E. and Rabiner, L.R., "Optimum FIR Digital Filter Implementations for Decimation, Interpolation, and Narrow-Band Filtering", IEEE Trans. Vol. ASSP-23, No. 5, October 1975, pp. 444-456.

9) Rabiner, L.R. and Crochiere, R.E., "A Novel Implementation for Narrow-Band FIR Digital Filters", IEEE Trans. Vol. ASSP-23, No. 5, October 1975, pp. 457-464.

10) Crochiere, R.E. and Rabiner, L.R., "Further Considerations in the Design of Decimators and Interpolators", IEEE Trans. Vol. ASSP-24, No. 4, August 1976, pp. 296-311.

11) Dickey, F.R. Jr., "Simple Algorithms for Approximating the Complex Envelope in Terms of Real Samples," Proc. IEEE, Vol. 59, No. 7, July 1971, pp. 1119-1121 (correspondence).

APPENDIX A

PROGRAMS

In this Appendix four computer program listings are given as mentioned in the main text. Block data subprogram C435 must be loaded along with C432 or C434. The two HP-67 programs have been coded so that they may be merged onto one magnetic card, if desired. The FORTRAN programs are also available in standard (FORTY) source-deck format.

```
10*C432  C432 ITERATIVE INTERPOLATION SUBROUTINE
20* JP COSTAS 12/13/78
30*USE WITH BLOCK DATA SUBPROGRAM C435
40*
50  SUBROUTINE LINT(VIN,FOUT)
60*
70*VIN-SINGLE INPUT DATA ITEM
80*FOUT-ARRAY FOR OUTPUT DATA STORAGE
90*
100*DIMENSIONED FOR MAX N OF 64
110*
120  COMMON /GP432/ SCOEF(512),NG(6,20)
130*
140  DIMENSION FOUT(1),HOLD(128,5),OUT(32,6),NCF(5),MP(5),
150& COEF(64,5),DELAY(128),USE(128),COEF2(64),
160& COMP(2),RE(2)
170*
180  EQUIVALENCE (DELAY(1),HOLD(1,1)),(USE(1),HOLD(1,2)),
190& (COEF2(1),COEF(1,1))
200*
210*
220*NT-INTERPOLATION TYPE 1-12 IN THIS VERSION
230*NG(6,NT)-ADDRESS-1 IN SCOEF OF COEFFICIENT DATA
240*NG(L,NT)-NUMBER OF COEFS AT LEVEL L
250*
260  OUT(1,1)=VIN
270  DO 1 L=1,NLEV
280  NCFL=NCF(L)
290  NCFM=NCFL-1
300  LENL=2*NCF(L)
310  LML=LENL-1
320  KMAX=2**(L-1)
330*KMAX IS NUM INPUT DATA VALUES AT THIS LEVEL
340  KK=0
350  DO 1 K=1,KMAX
360  MP(L)=MOD(MP(L)+1,LENL)
370  MPL=MP(L)
380  KK=KK+1
390  NEW=MOD(MPL+NCFM,LENL)
400  HOLD(NEW+1,L)=OUT(K,L)
410  BOUT=0.
420  M1=MPL
430  M2=MOD(MPL+LML,LENL)
440  DO 2 J=1,NCFL
450  M1=MOD(M1+LML,LENL)
460  M2=MOD(M2+1,LENL)
470 2 BOUT=BOUT+COEF(J,L)*(HOLD(M1+1,L)+HOLD(M2+1,L))
```

```
      OUT(KK,L+1)=BOUT
      KK=KK+1
    1 OUT(KK,L+1)=HOLD(MPL+1,L)
      IF(LEV.GT.5) GO TO 3
      DO 4 K=1,KA
    4 FOUT(K)=OUT(K,NA)
      RETURN
    3 KK=0
      DO 6 K=1,32
      BOUT=(OUT(K,6)-SAVE)/DEL
      DO 5 J=1,NDEL
      KK=KK+1
    5 FOUT(KK)=SAVE+FLOAT(J-1)*BOUT
    6 SAVE=OUT(K,6)
      RETURN
*
*
      ENTRY XLINT(NT,LEV,LAG)
*
*
*INITIALIZATION ENTRY FOR LINT
*NT-INTERPOLATION TYPE (1-12 IN THIS VERSION)
*LEV-NUM LEVELS OF INTERPOLATION
*EACH INPUT VALUE GENERATES 2**LEV OUTPUT VALUES
*LAG-DELAY THRU INTERPOLATOR IN TERMS OF OUTPUT SAMPLES
*
*
      SAVE=0.
      KK=NG(6,NT)
      DO 7 L=1,5
      MP(L)=-1
      M1=NG(L,NT)
      NCF(L)=M1
      DO 7 J=1,M1
      KK=KK+1
    7 COEF(J,L)=SCOEF(KK)/2.
      NLEV=MINO(LEV,5)
      DO 8 J=1,640
    8 HOLD(J,1)=0.
      DO 9 J=1,192
    9 OUT(J,1)=0.
      IF(LEV.GT.5) GO TO 10
      KA=2**NLEV
      NA=NLEV+1
      GO TO16
   10 NDEL=2**(LEV-5)
      DEL=FLOAT(NDEL)
```

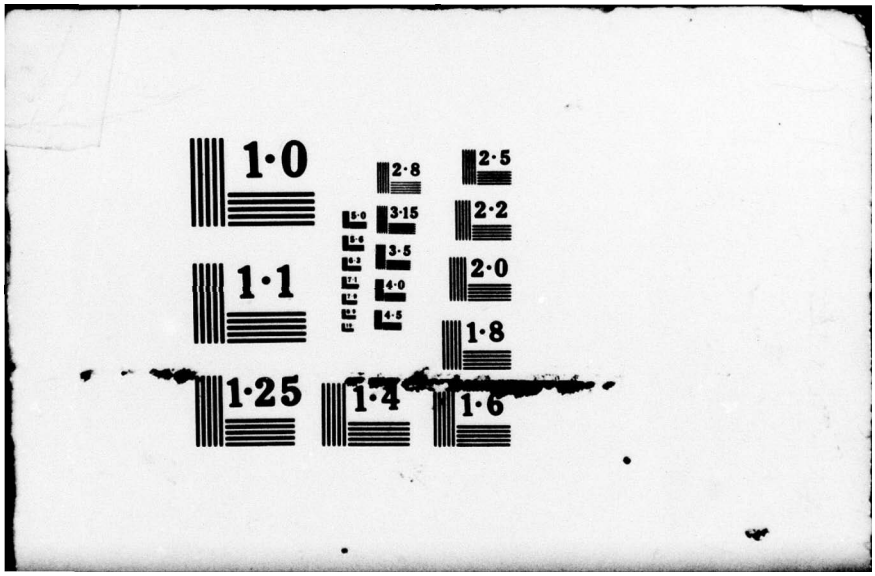
```
950 16 M1=LEV+1
960  LAG=1-2**LEV
970  DO 15 J=1,LEV
980  KK=1
990  IF(J.LE.5) KK=NG(J,NT)
1000 15 LAG=LAG+KK*2**(M1-J)
1010  RETURN
1020*
1030*
1040  ENTRY CONV(NT,LAGP)
1050*INITIALIZING ROUTINE FOR BOTH CORE AND RECO
1060*NT-INTERPOLATION TYPE 1-12 IN THIS VERSION
1070*LAGP-PROCESSOR DELAY IN TERMS OF OUTPUT SAMPLE PAIRS
1080  NCFL=NG(1,NT)
1090  LAGP=NCFL
1100  LENL=2*NCFL
1110  LML=LENL-1
1120  MPL=-1
1130  KK=NG(6,NT)
1140  DO 11 J=1,NCFL
1150  KK=KK+1
1160 11 COEF2(J)=SCOEF(KK)/2.
1170  DO 12 J=1,256
1180 12 HOLD(J,1)=0.
1190  FLIP=1.
1200  RETURN
1210*
1220*
1230  ENTRY CORE(COMP,RE)
1240*COMPLEX-TO-REAL TRANSFORMATION ENTRY
1250*COMP-LOCATION PAIR FOR COMPLEX INPUT
1260*RE-LOCATION PAIR FOR REAL OUTPUT
1270  MPL=MOD(MPL+1,LENL)
1280  RE(1)=DELAY(MPL+1)*FLIP
1290  FLIP=-FLIP
1300  NEW=MOD(MPL+NCFL,LENL)
1310  DELAY(NEW+1)=COMP(1)
1320  USE(NEW+1)=COMP(2)
1330  BOUT=0.
1340  M1=MOD(MPL+1,LENL)
1350  M2=MPL
1360  DO 13 J=1,NCFL
1370  M1=MOD(M1+LML,LENL)
1380  M2=MOD(M2+1,LENL)
1390 13 BOUT=BOUT+COEF2(J)*(USE(M1+1)+USE(M2+1))
1400  RE(2)=BOUT*FLIP
1410  RETURN
```

AD-A067 511 GENERAL ELECTRIC CO SYRACUSE N Y HEAVY MILITARY EQUI--ETC F/G 9/5
AN ITERATIVE INTERPOLATION TECHNIQUE USING FINITE IMPULSE RESPO--ETC(U)
FEB 79 J P COSTAS

UNCLASSIFIED R79EMH1 NL

2 OF 2
ADA
067511

END
DATE
FILMED
6-79
DDC


1·0
1·1
1·25
1·4
1·6
1·8
2·0
2·2
2·5
2·8
3·15
3·5
4·0
4·5

```
1420*
1430*
1440   ENTRY RECO(RE,COMP)
1450*RE-LOCATION PAIR FOR REAL INPUT
1460*COMP-LOCATION PAIR FOR COMPLEX OUTPUT
1470   MPL=MOD(MPL+1,LENL)
1480   COMP(2)=DELAY(MPL+1)
1490   NEW=MOD(MPL+NCFL,LENL)
1500   USE(NEW+1)=RE(1)*FLIP
1510   FLIP=-FLIP
1520   DELAY(NEW+1)=RE(2)*FLIP
1530   BOUT=0.
1540   M1=MOD(MPL+1,LENL)
1550   M2=MPL
1560   DO 14 J=1,NCFL
1570   M1=MOD(M1+LML,LENL)
1580   M2=MOD(M2+1,LENL)
1590 14 BOUT=BOUT+COEF2(J)*(USE(M1+1)+USE(M2+1))
1600   COMP(1)=BOUT
1610   RETURN
1620*
1630*
1640   END
```

```
10*C434  C434 ITERATIVE SAMPLING RATE DIVISION SUBROUTINE
20*JP COSTAS 2/7/79
30*USE WITH BLOCK DATA SUBPROGRAM C435
40*
50   SUBROUTINE LDIV(FIN,VOUT)
60*
70*FIN-INPUT DATA ARRAY
80*VOUT-SINGLE OUTPUT DATA VALUE
90*
100*DIMENSIONED FOR MAX N OF 64
110*
120   COMMON /GP432/ SCOEF(512),NG(6,20)
130*
140   DIMENSION FIN(1),DELAY(65,5),HOLD(128,5),OUT(32,6),NCF(5),
150& MP(5),MPD(5),COEF(64,5)
160*
170*LAST EFN IS 17
180*
190*
200*NT-INTERPOLATION TYPE 1-12 IN THIS VERSION
210*NG(6,NT)-ADDRESS-1 IN SCOEF OF COEFFICIENT DATA
220*NG(L,NT)-NUMBER OF COEFS AT LEVEL L
230*
240   IF(LEV.LE.5) GO TO 3
250*ENTER LINEAR INTERPOLATION PROCESSOR
260   SUM=FIN(1)
270   DO 6 J=1,NDELM
280 6 SUM=SUM+FLOAT(NDEL-J)/DEL*FIN(J+1)
290   OUT(1,1)=SUM+SAVE
300   MPL=1.
310   DO 11 K=2,32
320   MPL=MPL+NDEL
330   SUM=FIN(MPL)
340   DO 12 J=1,NDELM
350 12 SUM=SUM+FLOAT(NDEL-J)/DEL*(FIN(MPL+J)+FIN(MPL-J))
360 11 OUT(K,1)=SUM
370*NOW SET UP SAVE FOR NEXT CYCLE
380   SAVE=0.
390   DO 13 J=1,NDELM
400 13 SAVE=SAVE+FLOAT(J)/DEL*FIN(KB+J)
410   GO TO 14
420 3 DO 15 K=1,KA
430 15 OUT(K,1)=FIN(K)
440 14 CONTINUE
450*
460*
470*ENTER THE NORMAL CASCADE PROCESSOR
```

```
480   DO 1 L=1,NLEV
490   LP=NLEV+1-L
500   KMAX=2**LP
510*KMAX IS NUM INPUT DATA VALUES AT THIS LEVEL
520   NCFL=NCF(LP)
530   LENL=2*NCFL
540   LEND=NCFL+1
550   LML=LENL-1
560   KK=0
570*KK IS POINTER IN DATA OUTPUT ARRAY
580*
590   DO 1 K=1,KMAX,2
600   MP(L)=MOD(MP(L)+1,LENL)
610   MPL=MP(L)
620   MPD(L)=MOD(MPD(L)+1,LEND)
630   MPDL=MPD(L)
640   KK=KK+1
650   NEW=MOD(MPL+NCFL,LENL)
660   NEWD=MOD(MPDL+NCFL,LEND)
670   HOLD(NEW+1,L)=OUT(K,L)
680   DELAY(NEWD+1,L)=OUT(K+1,L)
690   BOUT=DELAY(MPDL+1,L)
700   M1=MOD(MPL+1,LENL)
710   M2=MPL
720   DO 2 J=1,NCFL
730   M1=MOD(M1+LML,LENL)
740   M2=MOD(M2+1,LENL)
750 2 BOUT=BOUT+COEF(J,LP)*(HOLD(M1+1,L)+HOLD(M2+1,L))
760 1 OUT(KK,L+1)=BOUT
770   VOUT=BOUT
780   RETURN
790*
800*
810   ENTRY XLDIV(NT,LEV,LAG)
820*
830*
840*INITIALIZING ENTRY FOR LDIV
850*NT-PROCESSOR TYPE (1-12 IN THIS VERSION) (INPUT DATA)
860*LEV-NUM LEVELS OF DIVISION (INPUT DATA)
870*LAG-PROCESSOR DELAY IN TERMS OF INPUT PULSES (OUTPUT DATA)
880*ONE OUTPUT VALUE REQUIRES 2**LEV INPUT VALUES
890*
900   SAVE=0.
910   KK=NG(6,NT)
920   DO 7 L=1,5
930   MP(L)=-1
940   MPD(L)=-1
```

```
      M1=NG(L,NT)
      NCF(L)=M1
      DO 7 J=1,M1
      KK=KK+1
    7 COEF(J,L)=SCOEF(KK)/2.
      NLEV=MING(LEV,5)
      DO 8 J=1,640
    8 HOLD(J,1)=0.
      DO 9 J=1,192
    9 OUT(J,1)=0.
      DO 16 J=1,325
   16 DELAY(J,1)=0.
      NDEL=2**(LEV-5)
      DEL=FLOAT(NDEL)
      KA=2**LEV
      KB=KA-NDEL+1
      NDELM=NDEL-1
      M1=LEV+1
      LAG=1-2**LEV
      DO 17 J=1,LEV
      KK=1
      IF(J.LE.5) KK=NG(J,NT)
   17 LAG=LAG+KK*2**(M1-J)
      RETURN
C
      END
```

```
10*C435  C435 BLOCK DATA FOR C432 AND C434
20*JP COSTAS 02/25/79
30*
40   BLOCK DATA
50*
60   COMMON /GP432/ SCOEF(512),NG(6,20)
70*
80*
90*NT-INTERPOLATION TYPE 1-12 IN THIS VERSION
100*NG(6,NT)-ADDRESS-1 IN SCOEF OF COEFFICIENT DATA
110*NG(L,NT)-NUMBER OF COEFS AT LEVEL L
120*
130   DATA (NG(J,1),J=1,6)/10,5,3,2,2,0/
140   DATA (NG(J,2),J=1,6)/7,4,2,2,1,22/
150   DATA (NG(J,3),J=1,6)/5,3,2,1,1,38/
160   DATA (NG(J,4),J=1,6)/3,2,1,1,1,50/
170   DATA (NG(J,5),J=1,6)/19,5,3,2,2,58/
180   DATA (NG(J,6),J=1,6)/14,4,2,2,1,89/
190   DATA (NG(J,7),J=1,6)/10,3,2,1,1,112/
200   DATA (NG(J,8),J=1,6)/6,2,1,1,1,129/
210   DATA (NG(J,9),J=1,6)/5,3,2,2,1,140/
220   DATA (NG(J,10),J=1,6)/4,2,2,1,1,153/
230   DATA (NG(J,11),J=1,6)/3,2,1,1,1,163/
240   DATA (NG(J,12),J=1,6)/2,1,1,1,1,171/
250*
260*
270*NT=1, 80%, -65DB
280  DATA(SCOEF(J),J=1,22)/1.26454,-.398901,.214047,-.128818,.0791232,
290&-.0475243,.0270681,-.0141908,.00651868,-.00254049,
300*
310&1.23918,-3.31141E-01,.124558,-4.07535E-02,8.75042E-03,
320&1.18469,-.215619,3.10496E-02,
330&1.13018,-.130329,
340&1.12628,-.12629/
350*
360*NT=2, 80%, -50DB
370  DATA(SCOEF(J),J=23,38)/1.26149,-.398195,.200846,-.112826,.0622564
380&,-.0316199,.0154772,
390*
400&1.23183,-.31273,.103554,-.025179,
410&1.14551,-.146007,
420&1.13018,-.130329,
430&1.00348/
440*
450*NT=3, 80%, -40DB
460  DATA(SCOEF(J),J=39,50)/1.25807,-.380468,.18634,-.0957456,.0545745
470*
```

```
480&,1.22058,-.285339,.0759446,
490&1.14551,-.148007,
500&1.01401,
510&1.00348/
520*
530*NT=4, 80%, -25DB
540 DATA(SCOEF(J),J=51,58)/1.25286,-.366063,.214951,
550*
560&1.20257,-.254076,
570&1.05764,
580&1.01401,
590&1.00348/
600*
610*NT=5, 90%, -65DB
620DATA(SCOEF(J),J=59,89)/1.27098,-.417671,.243546,-.166627,.122307,
630&-9.30066E-02,7.19947E-02,-5.61481E-02,4.38092E-02,-3.40215E-02,
640&2.61841E-02,-1.98932E-02,1.48724E-02,-1.08860E-02,7.74654E-03,
650&-5.3449E-03,3.51893E-03,-2.18409E-03,1.70604E-03,
660*
670&1.23918,-.331141,.124558,-.0407535,8.75042E-03,
680&1.18469,-.215619,3.10496E-02,
690&1.13018,-.130329,
700&1.12628,-.12629/
710*
720*NT=6, 90%, -50DB
730DATA(SCOEF(J),J=90,112)/1.27034,-.41577,.240448,-.162436,.117166,
740&-8.70883E-02,6.54880E-02,-4.92510E-02,3.67355E-02,-2.69847E-02,
750&1.93773E-02,-1.34576E-02,8.9452E-03,-6.51445E-03,
760*
770&1.23183,-.31273,.103554,-.025179,
780&1.14551,-.148007,
790&1.13018,-.130329,
800&1.00348/
810*
820*NT=7, 90%, -40DB
830DATA(SCOEF(J),J=113,129)/1.26955,-.413328,.23656,-.157144,
840&.110789,-.0797615,.0576061,-.0410165,.0287082,-.031823,
850*
860&1.22058,-.285339,.0759446,
870&1.14551,-.148007,
880&1.01401,
890&1.00348/
900*NT=8, 90%, -25DB
910DATA(SCOEF(J),J=130,140)/1.26844,-.409637,.230961,-.149662,
920&.103141,-.131136,
930*
940&1.20257,-.254076,
```

```
950&1.05764,
960&1.01401,
970&1.00345/
980*
990*NT=9,  60%,  -65DB
1000DATA(SCOEF(J),J=141,153)/1.23918,-.331141,.124558,-.0407535,
1010&.0067504 2,
1020*
1030&1.18469,-.215619,.0310496,
1040&1.13018,-.130329,
1050&1.12628,-.12629,
1060&1.00067/
1070*
1080*NT=10,  60%,  -50DB
1090DATA(SCOEF(J),J=154,163)/1.23183,-.31273,.103554,-.025179,
1100&1.14551,-.145007,
1110&1.13018,-.130329,
1120&1.00345,
1130&1.00067/
1140*
1150*NT=11,  60%,  -40DB
1160DATA(SCOEF(J),J=164,171)/1.22058,-.285339,.0759446,
1170*
1180&1.14551,-.145007,
1190&1.01401,
1200&1.00345,
1210&1.00067/
1220*
1230*NT=12,  60%,  -25DB
1240DATA(SCOEF(J),J=172,177)/1.20257,-.254076,
1250*
1260&1.05764,
1270&1.01401,
1280&1.00345,
1290&1.00067/
1300*
1310   END
```

PROGRAM 1-5-79 (HP-67), FOR USE WITH SECTION V, PAR. 5.3 EQUATIONS

1	(D)	fLBLD	
2		2	
3		5	
4		hSTI	
5	(5)	fLBL5	
6		fDSZ	
7		1	
8		STO(i)	
9		hRCI	
10		1	
11		0	
12		gx≠y	
13		GTO5	
14	(3)	fLBL3	
15		R/S	Enter $\{N_k\}$
16		STO(i)	
17		fISZ	
18		GTO3	
19	(C)	fLBLC	Enter L
20		STOO	
21		2	
22		hx ⇄ y	
23		hy^x	
24		ENT↑	
25		ENT↑	
26		1	
27		-	
28		STO2	
29		h R↓	
30		2	
31		x	
32		STO3	
33		CLX	
34		STO5	
35		STO6	
36		.	
37		5	
38		STO4	
39		9	
40		hSTI	
41	(2)	fLBL2	
42		fISZ	
43		RCL4	
44		2	
45		x	
46		STO4	
47		RCL(i)	$\{N_k\}$
48		x	
49		STO+6	O_i sum
50		RCL3	
51		2	
52		÷	
53		STO3	
54		RCL(i)	
55		x	
56		STO+5	N_f sum
57		hRCI	
58		9	
59		-	I - 9
60		RCLO	
61		gx≠y	
62		GTO2	
63		RCL5	
64		RCL2	
65		-	
66		R/S	Show N_f, D_i
67		RCL6	
68		R/S	Show O_i
69		÷	
70		R/S	Show N_f/O_i
71		RCL5	
72		2	
73		RCLO	
74		1	
75		-	
76		hy^x	
77		÷	
78		ENT↑	
79		fINT	
80		STO7	
81		gx=y ?	
82		GTO4	
83		1	
84		STO+7	
85	(4)	fLBL4	
86		RCL7	
87		R/S	Show N_c

PROGRAM 1-11-79 (HP-67) FOR USE WITH SECTION V, PAR. 5.4 PROCEDURES

1	(A)	fLBLA	
2		STOO	
3		fLN	
4		2	
5		fLN	
6		÷	
7		fINT	
8	(1)	fLBL1	
9		ENT↑	
10		1	
11		+	
12		STO2	
13		R/S	Show L
14		ENT↑	
15		2	
16		hx $\rightleftarrows$ y	
17		h y^x	
18		RCL0	
19		÷	
20		STO4	
21		ENT↑	
22		fINT	
23		STO3	
24		ENT↑	
25		h 1/x	
26		2	
27		+	
28		hR↓	
29		-	
30		hR↑	
31		X	
32		fINT	
33		RCL3	
34		+	
35		STO1	
36		RCL4	
37		hx $\rightleftarrows$ y	
38		÷	
39		1	
40		-	
41		hABS	
42		EEX	
43		2	
44		X	
45		f - x -	Show |Δ|%
46		RCL1	
47		f - x -	Show J
48		2	
49		RCL2	
50		hy^x	
51		RCL1	
52		÷	
53		f - x -	Show $\hat{K}$
54		RCL2	
55		GTO1	